Optimized analysis of variable chlorophyll fluorescence in algal physiology under stress conditions : measuring nothing with confidence

Nicole Bobco
California State University, Monterey Bay

OPTIMIZED ANALYSIS OF VARIABLE CHLOROPHYLL FLUORESCENCE IN ALGAL PHYSIOLOGY UNDER STRESS CONDITIONS: MEASURING NOTHING WITH CONFIDENCE

A Thesis

Presented to the Faculty of the

Moss Landing Marine Laboratories

California State University Monterey Bay

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

in

Marine Science

---

by

Nicole Bobco

Spring 2014

CALIFORNIA STATE UNIVERSITY MONTEREY BAY

The Undersigned Faculty Committee Approves the

Thesis of Nicole Bobco:

OPTIMIZED ANALYSIS OF VARIABLE CHLOROPHYLL FLUORESCENCE IN ALGAL PHYSIOLOGY UNDER STRESS CONDITIONS: MEASURING NOTHING WITH CONFIDENCE

Nicholas Welschmeyer, Chair
Moss Landing Marine Laboratories

E Sh

Erika McPhee-Shaw
Moss Landing Marine Laboratories

G. Jason Smith
Moss Landing Marine Laboratories

Marsha Moroh

Marsha Moroh, Dean
College of Science, Media Arts, and Technology

5/12/14

Approval Date

ABSTRACT

Optimized Analysis of Variable Chlorophyll Fluorescence in Algal Physiology Under Stress Conditions: Measuring Nothing With Confidence

by

Nicole Bobco

Masters of Science in Marine Science

California State University Monterey Bay, 2014

The detection and determination of viable phytoplankton are fundamental assays in oceanographic and limnological fields. Assessing phytoplankton physiology allows for insight into how cells might respond to different environmental or physical stressors, making it an important subject for ecological research and applied sciences, where the physiological status of photautotrophs must be known. In particular, maritime operations have been identified as major vectors in the spread of aquatic invasive species through ballast water transport. To mitigate this environmental hazard, shipborne systems are required to effectively sterilize ballast water to comply with emerging regulations regarding the transport of ballast water. Common oceanographic tools must now be reassessed for their capacity to quantify the treatment based reduction in viable biomass for organisms, like phytoplankton, to determine efficacy and compliance of treatment systems with ballast water regulations. In order to meet the time-cost demands of maritime operations, ballast treatment compliance assessment methods need to provide rapid test results on site in contrast to gold standard methods like microscopic enumeration. Pulse Amplitude Modulated (PAM) fluorometry offers one method to assess the phytoplankton component of ballast water. This technology, exemplified by the WALZ WATER-PAM, measures chlorophyll fluorescence kinetics to provide various indices (e.g. F_v/F_m) of photosystem activity, a proxy of cell viability or phytoplankton health. Unfortunately this method is optimized for high biomass suspensions and distinguishing low concentrations of viable cells from inactive cell samples, as required for ballast treatment monitoring applications, challenges the sensitivity limits of the current instrumentation package. This thesis describes the development and testing of a series of custom algorithms developed in MATLAB to enable robust assessment of PAM rapid light curves trends in order to quantify the absence of biological activity in treated ballast water samples. The series of algorithms provide a simple user interface and has practical applications in a wide range of academic as well as regulatory applications.

TABLE OF CONTENTS

LIST OF TABLES

PAGE

LIST OF FIGURES

ACKNOWLEDGEMENTS

I would like to express my deepest appreciation to my advisor Dr. Nicholas Welschmeyer for his expert guidance and support as a student in his lab during the pursuit of my degree. In addition, I would also like to give a special thanks to my other committee members, Dr. Erika McPhee-Shaw for her wisdom, counsel and encouragement and to Dr. G. Jason Smith for his thoughtful questions and comments throughout this process. To my family and friends, thank you so much for your continued support during this journey. Finally, I would like to thank the entire MLML community for making the time spent at Moss Landing Marine Labs a memorable and enjoyable experience.

INTRODUCTION

The Invasive Species Problem

Many of the aquatic species invasions that have occurred over the past 20 years appear to be related to ballast water discharge (Carlton and Geller 1993). The purpose of ballast water on ships is to achieve balance and stability when loading and transporting cargo, a process that has been used regularly since the late 1800s (Hallegraeff and Bolch 1991). Initial ballast water intake contains many planktonic organisms, which are consequently released at various ports of destination, potentially thousands of miles away (Hallegraeff and Bolch 1992). Some discharged organisms have the potential to become invasive species and therefore threaten, not only ecological stability, diversity and abundance of native taxa, but commercial activities as well (Reavie *et al.* 2010).

The unmanaged discharge of ballast tanks has been recognized as a global issue. Stringent regulations have been put in place by various organizations around the world and require that the number of organisms, according to specific size classes (<10μm, 10-50μm, >50μm), not exceed specified limits (International Maritime Organization/IMO 2004; U.S. Coast Guard 2009; U.S. Environmental Protection Agency 2011a, b). For example, the biological criteria for a successful ballast water treatment, as stipulated in IMO Regulation D-2, organisms falling between a ≥10 and <50 μm in minimum dimension size class must not exceed 10 viable organisms per mL of water (IMO, 2004). However, due to the diversity of the taxa within this size class (primarily phytoplankton with some protists and zooplankton), there is no single method that can irrefutably provide the exact numeric concentration of *all* organisms (Dobroski *et al.* 2000). Therefore, alternative sensitive and convenient methods for analysis are now desired for future compliance testing purposes. The challenge for ballast water compliance monitoring is to obtain confidence that lack of an analytical signal or low level signal is real rather than an artifact of the analytical techniques' limit of detection.

The objective of the present research was to assess and improve the Pulse Amplitude Modulated Fluorometer (PAM) measurements of variable chlorophyll fluorescence, a bio-optical indicator of physiological status in phytoplankton. This work sought to make PAM measurements more sensitive, reliable and efficient for evaluating ballast water treatment system efficacy for the photoautotrophic organisms within the ≥ 10 and <50 µm size class as well as other for oceanographic purposes.

Current Methods to Measure Viable Biomass

A range of techniques to measure viable aquatic biomass of organisms in the ≥ 10 and <50 µm size class currently exist; however each method possesses its own advantages and drawbacks for measuring photoautotrophic viability.

CHLOROPHYLL *A* AND FLUOROMETERS

Chlorophyll *a* is the primary light-harvesting pigment involved in photosynthesis, and is present in all oxygenic phytoplankton. Chlorophyll *a* is fluorescent, meaning the pigment strongly absorbs high energy light (blue) and emits fluorescence at a lower energy wavelength beyond 660nm (red) (Schreiber et al., 1998). Phytoplankton concentrations are frequently inferred from bulk measurements, such as total chlorophyll concentrations (Bidigare *et al.* 1986; Welschmeyer 1994) or whole-cell fluorescence since the intensity of the fluorescence can be quantified.

Fluorometers take advantage of fluorescent properties of certain compounds (like chlorophyll), in order to quantify the concentration of the sample. They have the ability to select a specific excitation wavelength and observe the emission of a narrow wavelength range or allow excitation at a range of wavelengths in a scan to determine the intensity vs the wavelength. Some examples include reading the intensity *in situ* with a fluorometer or extracting chlorophyll in an organic solvent and quantifying the *in vitro* intensity with a spectrophotometer (Townsend 2012). The benefit of using chlorophyll *a* is that the concentration per unit phytoplankton biomass is relatively constant, so chlorophyll *a* readings are a good proxy for phytoplankton concentration. However in situ measurements are confounded by the organism's biology resulting in species-specific and environmentally dependent chlorophyll fluorescence yields. Additionally, even in dead cells chlorophyll *a* remains intact and fluorescent for long periods of time, therefore chlorophyll *a* fluorescence by itself is not a reliable indicator of viable algal biomass.

Chlorophyll-Based Most Probable Number (MPN)

Grow out techniques are commonly used to quantify the presence of live organisms since viable cells exhibit growth over time. For example, photoautotrophic growth can be measured from long-term incubation experiments using whole-cell chlorophyll fluorescence and quantified using the Most Probable Number (MPN) method. The experiments consist of a culture array composed of a series of dilutions of sample water, in replicate, and measured for growth after a set period of time. The change in measured whole-cell chlorophyll fluorescence over time is applied to an algorithm, based on the number of replicates exhibiting growth at each dilution, to calculate the most probable initial concentration of live cells. However, many aquatic photoautrophic and even heterotrophic microorganisms from natural samples are difficult to grow in a laboratory environment, making estimates of viable biomass conservative at best.

Microscopic Enumeration

Enumerating concentrations of viable phytoplankton generally consists of concentrating, tagging and manually counting a portion of the concentrated sample under a microscope. In the natural environment, phytoplankters are often unevenly and sparsely distributed, making concentration necessary for statistical validity. However, sampling issues such as net clogging and poor retention of intended populations can lead to errors in estimation of concentrations (First and Drake 2012). Furthermore, viability tags or markers are often used to stain the sample in order to differentiate the live and dead cells since most phytoplankton species are non-motile and therefore indistinguishable. These markers are usually chromogenic molecules, which can have fluorescence induced either when enzyme activity is present to indicate live cells (fluorescein diacetate (FDA), cmFDA, calcein AM) or when they bind to DNA in cells with damaged plasma membranes/dead cells (Sytox Green). However, time is a major constraint on the counting process since the markers, especially fluorescein, start to leak out of cells over a relatively short period of time (less than a half hour) causing ambiguity between the live and dead cells (Rotman and Papermaster 1966). Manual counts are also prone to error due to the subjectivity of the counter and the tedious nature of counting enough of the concentrated samples for statistical validity. Flow Cytometric Enumeration

In combination with fluorescence staining, flow cytometry allows for quantitative analysis of viable algal cells. As in microscopic enumeration, stains or viability markers (FDA, Styox, etc..) are commonly used to tag cells since the fluorescence is easily detected and measured by the flow cytometer (Breeuwer *et al.* 1995). The stained sample enters the flow cytometer, is forced into a thin stream of individual cells, and the fluorescence and size of the cell is detected by the instrument (Shapiro 2003). Cells tagged with the stain have a higher fluorescence signature than those without, allowing for differentiation between live and dead cells. Even though this automated system removes the human subjectivity error, it is still prone to the marker leakage issue leading to false positive counts. Furthermore, the instrument is limited to cell sizes <70 μm, since the aperture is clogged easily.

Variable Fluorescence

Variable fluorescence techniques have emerged as promising tools for assessing algal physiology *in situ* and delivering real time results. The two most common methods for inducing and measuring variable fluorescence were developed during the mid 1980s. They include pulse amplitude modulated (PAM) fluorometry (Falkowski et al., 1986, Schrieber et al. 1986, 1995) and the pump and probe technique, later modified to the fast repetition rate (FRR) fluorometer (Kolber et al. 1998). Since then, numerous fluorometers have been developed for measuring variable fluorescence. For example, the Pulse Amplitude Modulated fluorometer (PAM, WALZ GmbH, Effeltrich, Germany), Fluorescence Monitoring System (FMS, Hansatech Ltd, Kingy's Lynn, England), Fast Repetition Rate Fluorometer (FRRF; Chelsea Instruments Ltd., West Molesey, England), and Fluorescence Induction and Relaxation (FIRe, Satlantic, Halifax, Canada). The principle of operation shared by all of these modulated fluorometers is described below.

Light Budget in Photosynthetic Cell

Light energy absorbed by chlorophyll molecules in a photosynthetic cell may follow one of three paths: 1) it can be used to drive photosynthesis (photochemistry), 2) excess absorbed light energy can be dissipated as non-radiative dissipation (heat) and/or 3) re-emitted as long-wavelength fluorescence (Maxwell and Johnson 2000). Figure 1 illustrates the possible paths of absorbed light in the form of photons for a photosynthetic cell.

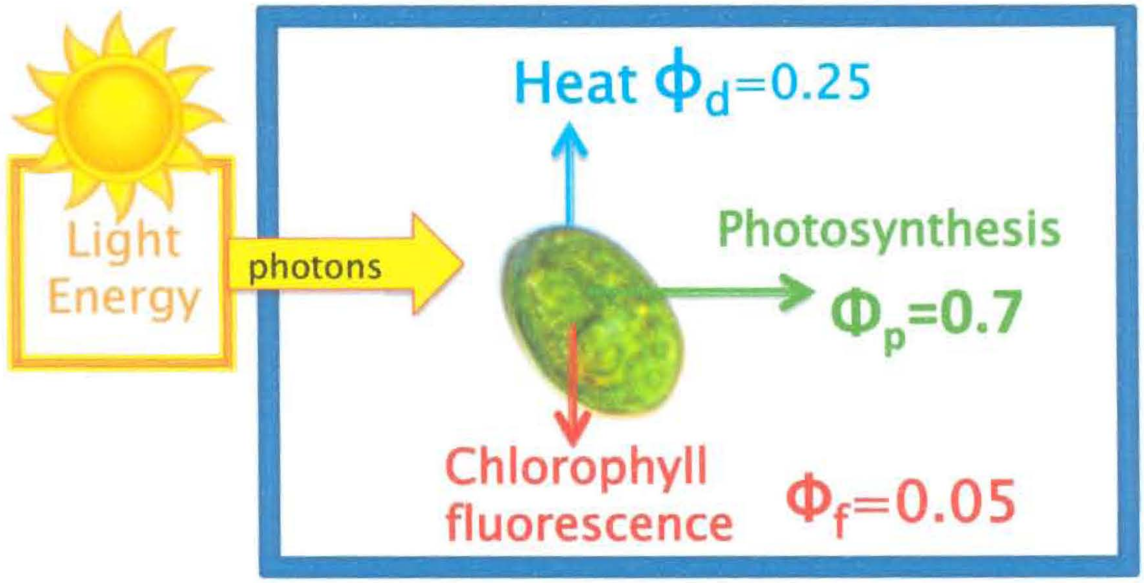

Figure 1. Path of light in a photosynthetic cell, aka. light budget

When a photosynthetic cell is in the dark for an extended period of time (aka. dark-adapted), even for 5 minutes, most if not all of its light harvesting pigments are ready to accept photons and transfer them to the reaction center chlorophyll (photo-system II) to do photosynthesis. However, there is a 'process time' associated with that transfer of energy because the cell can only deal with so much light at once. If there is a lot of energy in the form of light incident on the cell, more than can possibly be processed through the energy conversion of photosynthesis, the excess light can be harmful. The cell has to dissipate that energy in some other way. These other paths are 1) non-radiative decay (primarily in the form of heat) and 2) natural chlorophyll fluorescence. Each 'fate' has an associated quantum efficiency denoted by the Greek letter phi ϕ (with corresponding subscripts), which represents a ratio of photons performing an action to the number of photons absorbed (Figure 1). These processes, photosynthesis, heat dissipation and fluorescence, are competitive because any increase in the efficiency of one will result in a proportional decrease in the others. They therefore create a light energy budget: energy going in has to equal the energy going out. For example, if photosynthesis operates normally, it uses about 70 percent of the photons absorbed and fluorescence is low (around 5 percent). If photosynthesis is affected in some way, such as when reaction centers are shut down even temporarily, fluorescence will increase proportionally. Special modulated fluorometers can quantify this change in fluorescence, so we can therefore use it as a proxy for photosynthetic performance.

Variable Fluorescence

Upon illumination of a dark-adapted photosynthetic organism, there is a quick rise in fluorescence from photo-system II (PSII) followed by a slow decline. This phenomena is termed the Kautsky effect, first observed by Kautsky and Hirsch 1931. The increase is due to the PSII reaction centers being in a "closed" state, i.e. unable to accept additional electrons (converted light energy) because the electron acceptors farther down the transport system of PSII have not yet been processed. The level of fluorescence increases because closed reaction centers reduce the overall efficiency of photochemistry. At any given time when a photoautrophic cell is exposed to light, a portion of these reaction centers will be opened and closed. Live cells therefore produce variations in fluorescence emission in response to variations in excitation energy and proportion of absorbed light transferred to PSII (Suggett *et al.* 2005). Subsequently, the fluorescence will decrease over a few minutes due to quenching mechanisms (*see page 9*).

Measuring Variable Fluorescence

Variable fluorescence is induced by modulated flashes of light. A typical trace of modulated fluorescence is illustrated in Figure 2 with the corresponding parameters described further in Table 1. The baseline minima parameter, F_o, is defined as the minimum level of fluorescence with maximum opening of PSII centers in the dark-adapted state. This measurement is initiated by turning on the measuring light (ML) in the dark and while this light theoretically flashes at a frequency and intensity too low to activate photosynthesis, the frequency of the measuring light can be varied. The maximum level of fluorescence in the dark-adapted state, parameter F_m, is measured after a saturating pulse (SP) of high intensity light is applied (Maxwell and Johnson 2000). The ratio of the difference between the minima (F_o) and maximum (F_m) to the maximum fluorescence (in the dark-adapted state), gives the optimal quantum yield, F_v/F_m. Once the actinic light (AL) is turned on and further saturation flashes are applied, the minimum fluorescence (or steady state fluorescence, F_t) and the maximum fluorescence in the light-adapted state (F'_m) can be measured, respectively. In presence of the actinic light (aka. light-adapted state), the quantum yield or effective quantum efficiency of PSII is defined as $(F_v/F_m)'$ (Maxwell and Johnson 2000). Previous studies have shown that measurements of the maximum quantum efficiency of photosystem II (F_v/F_m) are highly sensitive to various conditions of the cells. This ratio gives the largest fluorescent range and allows for direct comparison to other cell samples because it is a sensitive indicator of photosynthetic performance. Healthy cells are expected to have dark-adapted F_v/F_m values between 0.5-0.8. However, stress will drive the value down, with dead or stressed cells often characterized by F_v/F_m values approaching zero (Maxwell and Johnson 2000). Examples of stress include exposure to biocides, heat, nutrient depletion and photoinhibition (White and Critchley 1999).

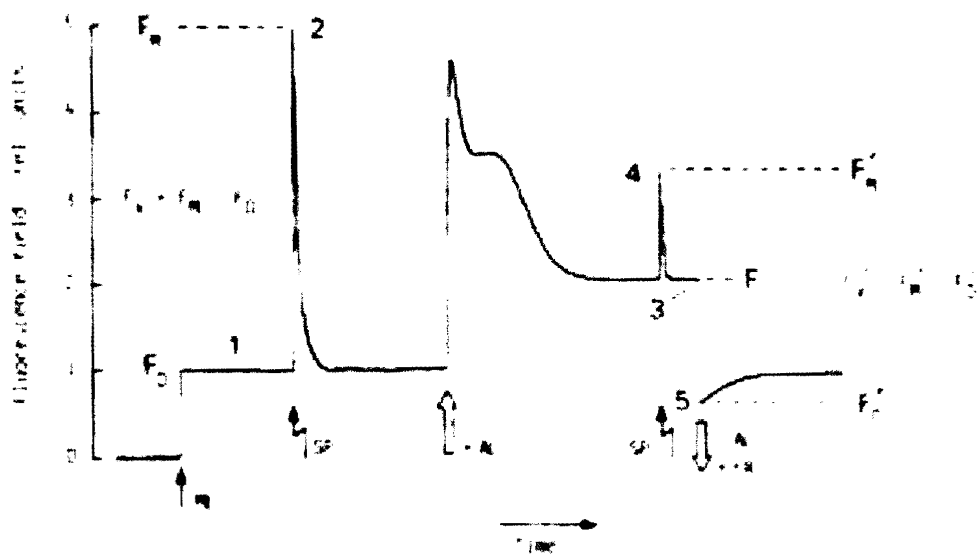

Figure 2. Fluorescence trace with parameters (van Kooten and Snel 1990).

Table 1. Fluorescence Parameters (van Kooten and Snel 1990).

Parameter	Definition	Units
F_o	Minimum fluorescence (dark-adapted state)	Instrument Specific
F_t or F_s	Minimum fluorescence (light-adapted state)	Instrument Specific
F_m	Maximum fluorescence (dark-adapted state)	Instrument Specific
F'_m	Maximum fluorescence (light-adapted state)	Instrument Specific
$F_v = F_m - F_o$	Variable fluorescence (dark-adapted state)	Dimensionless
$F'_v = F'_m - F_s$	Variable fluorescence (light-adapted state)	Dimensionless
$\frac{F_v}{F_m} = \frac{F_m - F_o}{F_m}$	Optimal quantum yield (dark-adapted state)	Dimensionless
$\frac{F'_v}{F'_m} = \left(\frac{F_v}{F_m}\right) = \frac{F'_m - F_s}{F'_m}$	Effective quantum yield (light-adapted state)	Dimensionless

Quenching Parameters

When fluorescence is reduced, either by increased energy usage for photosynthetic reactions or dissipating heat, the pathways are known as quenching mechanisms (Table 2, *below*). Photochemical quenching (qP) is a term that indicates more energy is being used for photochemical reactions. This is measured with the equation in Table 2; it is a ratio comparing the difference between the maxima and steady state minima during actinic light exposure (F'_m-F_s) to the difference between the maxima and steady state minima after actinic light exposure (F'_m-F'_o). This equation gives an indication of the proportion of the PSII reaction centers that are open or oxidized (Q_a acceptors available) (Maxwell and Johnson 2000, Mackey *et al.* 2008). Therefore, qP can alternatively be defined as: the fraction of oxidized Q_a acceptors available, the relative oxidation state of Q_a, or the "openness" of the reaction centers. Photochemical quenching can also be defined in terms of the proportion of closed PS II reaction centers given as 1-qP.

Table 2. Quenching Parameters (Maxwell and Johnson 2000, van Kooten and Snel 1990).

Quenching Parameter	Definition
$qP = \frac{F'_m - F_s}{F'_m - F'_o}$	Photochemical quenching; proportion of open PS II reaction centers.
$1 - qP$	Proportion of closed PS II reaction centers.
$qN = \frac{F_m - F'_m}{F_m - F'_o} = 1 - \left(\frac{F'_m - F'_o}{F_m - F_o} \right)$	Non-photochemical quenching.
$NPQ = \frac{F_m - F'_m}{F'_m}$	Non-photochemical quenching (alternative to above).

Non-photochemical quenching (NPQ, qN) measures a change in the efficiency of heat dissipation, relative to the dark-adapted state. This increase can occur as a result of either processes that either protect the cell from light induced damage or from the damage itself (Maxwell and Johnson 2000). Typically NPQ is used more than qN because the equation for NPQ incorporates information about the ratio of a change in maximum fluorescence values (F_m and F'_m) instead of F'_o, a difficult parameter to quantify, which is in the qN equation (Table 2). There are several components that contribute to non-photochemical quenching; processes that are based on the reaction times ranging from a few minutes to several hours.

Table 3. Non-Photochemical Quenching Parameters (Kropuenske *et al.* 2009,Maxwell and Johnson 2000).

Parameter	Definition
$qE = (F_{mR} - F'_m) - (F_m - F_o)$	Photochemical quenching; proportion of open PS II reaction centers.
qT	State transitional quenching; fast relaxing.
$qI = \frac{(F_m - F_{mR})}{(F_m - F_o)} =$	Photoinhibitory quenching; slow relaxing[1].

NPQ is a result of both fast-acting quenching (qE and qT) and more slowly relaxing quenching (qI). Fast acting energy dependent quenching (qE) is thought to be a major contributor to NPQ and is closely correlated with thermal deactivation of excess light energy through the involvement of xanthophyll cycle pigments, essential in protecting the cell from light damage (Maxwell and Johnson 2000; Kropuenske *et al.* 2009). State transitional quenching (qT) is thought to be important in balancing the distribution of light energy between PS I and PSII at low light, but generally makes a small contribution to overall quenching. However, it is a difficult quenching mechanism to quantify, so is usually ignored due to its minimal contribution (Maxwell and Johnson 2000). Photoinhibitory quenching (qI) is slow relaxing (hours or days) and has been attributed to damage or down regulation of PS II reaction centers as well as to conformational changes in the thylakoid structure (Kropuenske *et al.* 2009). It results in the decrease in the F_v/F_m parameter, which has been previously been found to correlate linearly with a decrease of the measured optimal quantum yield of photosynthesis.

Differences between PAM and FRR Fluorometers

Both pulse amplitude modulated (PAM) and fast repetition rate (FRR) fluorometers use the same definitions and a similar physiological principal for initiating a fluorescent response; however there are inherent differences between the excitation processes and resulting measurements of maximum fluorescence. FRR fluorometers deliver a rapid sequence of flashes on the order of approximately 2700 μmol photons m^{-2} s^{-1} over a short duration of 150-400 μs to obtain maximum fluorescence (F_m)(Figure 3B). This results in a simultaneous, single turnover (ST) of most PSII reaction centers. (Suggett et al. 2005). PAM fluorometers induce multiple photochemical turnovers (MT) of individual PSII reaction centers by using a prolonged duration (~0.8 s) of saturating light to determine F_m (Figure 3A). The consequent measurement difference is the MT method provides consistently higher F_m values than the ST technique. While both methods fully reduce the pool of acceptors, MT is the only method that fully reduces plastoquinone to plastoquinol. Plastoquinone is a quencher of chlorophyll fluorescence, so the maximum fluorescence measured using MT increases as plastoquinone is reduced to plastoquinol (Vernotte et al. 1979).

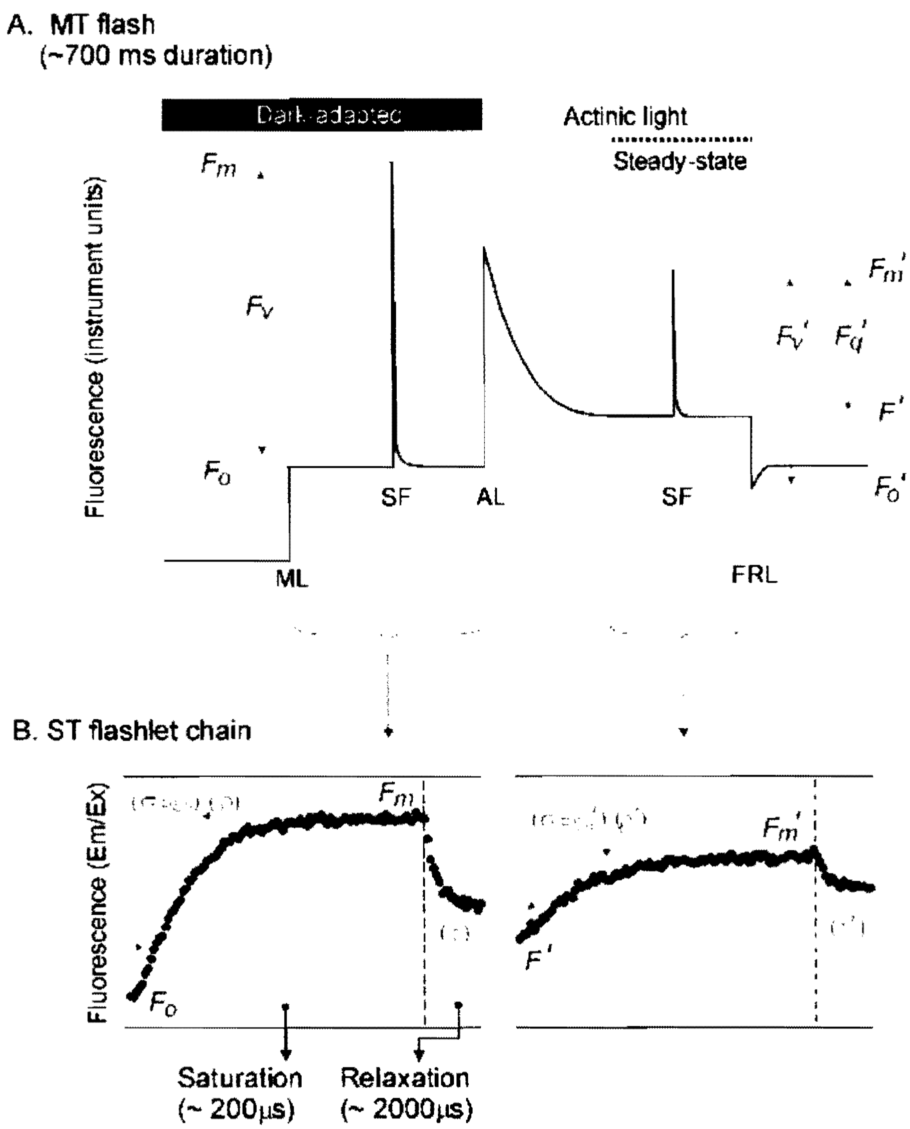

Figure 3. Comparison of variable fluorescence traces (Suggett et al. 2005): Multiple turnover (A) and single turnover (B) flashes.

Rapid Light Curves

Most modulated fluorometers can be programmed to create artificial irradiance experiments (e.g., rapid light curves) whereby physiological photosynthetic responses can be evaluated under well-controlled levels of actinic light. These light curve experiments essentially put the cells through a treadmill of increasing light levels in order to test their ability to deal with excess light and allow for the option to recover once the actinic light is turned off (Figure 4, #3). They provide estimates of non-photochemical quench, maximum levels of irradiance-saturated photosynthesis and recovery rates from irradiance-stressed conditions to dark relaxation (White and Critchley 1999). Previous studies have used these functions to better understand physiological responses to environmental conditions (Mackey et al. 2008, Kropuenske et al. 2009) and the quantitative models have been developed to budget the fate of absorbed light energy under standard test conditions (Ma *et al.* 2011, Bailey *et al.* 2008, Hendrickson *et al.* 2004).

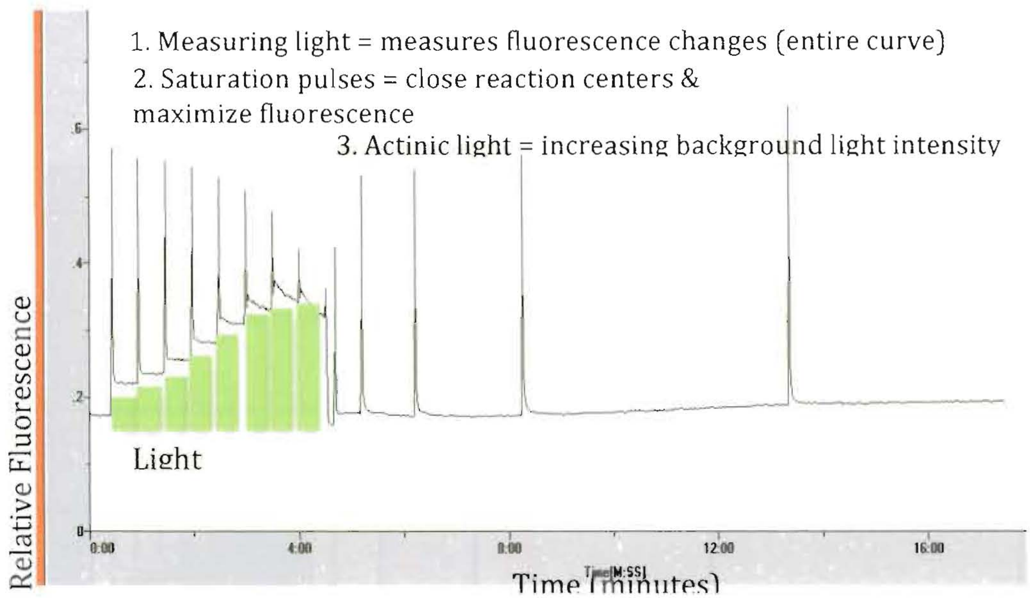

Figure 4. Programmed rapid light curve fluorescence trace with defined light sources: *Tetraselmis* sp. culture

The advantage for using a PAM fluorometer instead of an FRR fluorometer, is the ability to run rapid light curves (RLCs). It is basically is a sequence of F_v/F_m measurements strung together in a systematic fashion, hence the series of peaks in Figure 4. There are unique features in a RLC because it requires 3 sources of light: the entire fluorescence trace is generated by the measuring light measuring the changes in fluorescence; the large peaks reflect the fluorescence trace of the measuring light when an additional saturation pulse is applied; and the step-wise increase in the first half of the curve is due to the third light source, the actinic light, which causes the plant to photosynthesize. The actinic light systematically increases in intensity from dawn to midday sun type values as represented by the green bars in Figure 4. The only way to measure fluorescence of the measuring light is to modulate all three light sources.

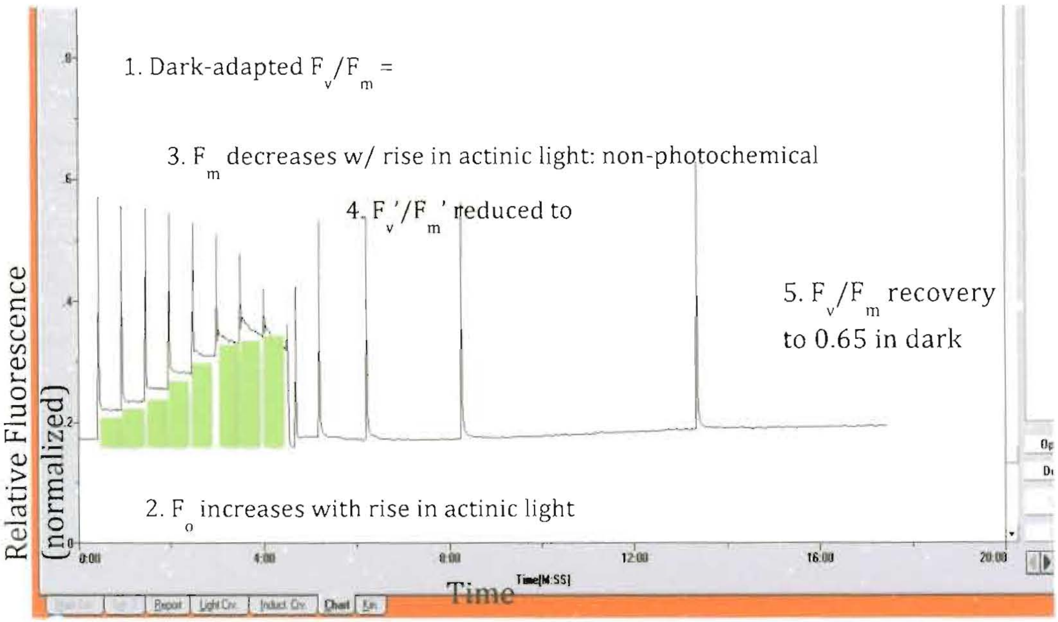

Figure 5. Programmed rapid light curve fluorescence trace highlighting trends: *Tetraselmis* sp. culture.

Performing a RLC experiment with a recovery period means complementary measurements regarding physiology are collected over a period of time (Schreiber et al., 1998). The starting point for a new RLC involves measuring the dark-adapted F_v/F_m ratio with a saturation pulse, which sets a baseline for health. After that initial pulse, the cells are exposed to a higher intensity actinic light level for about 30 seconds followed by a saturation pulse to measure the change in F_v/F_m. This pattern continues for about 5 minutes after which the actinic light is then turned off. Healthy cells are able to recover, usually within minutes, indicated by the final saturation pulse F_v/F_m approaching the initial dark-adapted F_v/F_m ratio. The trends formed as a result of the programmed RLC include: a response to increasing actinic light in the form of the minimum fluorescence, F_o increasing during the first half of the RLC; the maximum level F_m decreasing due to non-photochemical quenching; and the combination of the increase in F_t and decrease in F_m lead to a low F_v/F_m ratio at the end of the actinic light portion of the RLC. When the trends are combined, this approach can potentially lead to more certainty about whether a signal is present or absent from a low chlorophyll sample.

Purpose

The instruments that provide these fluorescent measuring capabilities range widely in complexity, cost and sensitivity. The most widely used instrument for aquatic samples, the WALZ manufactured WATER-PAM fluorometer, has a specified operational sensitivity to ca. 0.5 μg Chl/L. This is adequate for productive coastal waters, but is limited in open ocean water (with low concentrations of live cells, <0.05 μg Chl/L) or in successfully treated ballast water (with substantially reduced levels of chlorophyll concentration). In order to use the WALZ PAM fluorometer as a potential alternative for ballast water compliance testing, it is crucial to understand and quantify the minimum 'live' signal that can be reproducibly determined from the WALZ WinControl software chart data. Furthermore, it is important to create a method that can incorporate the WALZ instrument data and analyze rapid light curves in a timely manner. The objective of this thesis is therefore twofold: first to develop a series of algorithms in MATLAB to automate file processing and analyze rapid light curve trends for viability indicators; and secondly, to define minimum viability thresholds for the analyzed trends in order to quantify the absence of biological activity. The goal is that the series of algorithms produced as a result of this thesis will improve the PAM rapid light curve experiments by increasing the sensitivity and confidence in the presumed level of detection for phytoplankton viability.

Thesis Questions

1. What parameters indicate a biological response in a rapid light curve?
    a. Is the measurement of F_v/F_m (initial, dark-adapted) greater than zero?
    b. Is there a threshold for peak detection e.g. does each peak have a specific signal to noise ratio or standard deviation value that must be met to qualify as a true signal? Does it differ from signals given by water without any presumed live cells (e.g. Milli-Q or filtered sea water)?
    c. Is there a light induced response to F_t (baseline fluorescence)? e.g. is there a general baseline trend or is it sample specific?
2. What criteria must be met in order to differentiate between a healthy sample and a stressed or non-functional sample?

a. Does inventing a calculation (using two different approaches) that encompasses all the fluorescence points in a rapid light curve yield a numerical value that 'bulk' characterizes result in a the overall 'health' of the sample?

b. Do a combination of factors including 'live response' characteristics from the RLC along with signal to noise ratio or the score value give more confidence for determining cell viability?

3. Do the calculated F_v/F_m ratios using the series of algorithms correspond to WALZ's calculated F_v/F_m ratios?
4. Does concentrating treated samples yield a higher F_v/F_m ratio than the unconcentrated treatment samples?

METHODS

Rapid Light Curve Experiments

The rapid light curve (RLC) experiments were performed over the course of a testing phase for an experimental UV ballast water sterilization system aboard the T.S. Golden Bear using sample water from both control and treated tanks. The software used to run the RLC experiments was the WALZ Win-Control v2.8. The measuring light for all data collected was set at the factory setting of 3 (options vary from 1-12), which has an approximate flash rate of 18 times per second. During the actual saturation flash, the instrument flashed the light at a frequency of 20kHz (~20,000 times) in a 0.8 s interval and recorded the average maximum value in order to calculate F_v/F_m at each step in the RLC, according to the manufacturer. The photosynthetically active radiation, PAR, in the WATER-PAM comes from an internal LED light source to fully control the photon flux density. Each intensity setting of actinic light corresponds to a defined LED intensity, which in the given optical geometry, results in a defined PAR-value. A list of the PAR-values as a function of intensity settings is stored in the instrument (Internal PAR-List). The Internal PAR-List is based on empirical PAR sensor calibrations for each PAM optical unit. Each RLC experiment was carried out according to the following protocol, unless otherwise stated.

Protocol

The water samples used for the ballast water experiments were from the San Francisco Bay across different seasons. For each ballast 'test cycle', unaltered bay water was drawn into the ballst tanks (control), treated once with UV (half-dose), and then treated a second time (double-dose treatment) upon release into the bay. For each stage of the process, replicate samples were slowly drawn from each ballast tank into polypropylene carboys so each subsample reflected a time-integrated, representative sample of the full ballasing operation, which lasted ca. 2 hours. The samples used for analysis include 50 control, 50 treatment, and 19 half-dose samples that are paired from 25 different test cycles.

Once at the lab a 2.5mL subsample was drawn from each well mixed dark-adapted carboy, transferred into a round quartz cuvette, and placed in the dark measuring chamber. Then the instrument was prepped following the steps below before running a rapid light curve (RLC) experiment.

1. ***Check sample minimum fluorescence***
2. ***Set PM gain***
3. ***Blank with Milli-Q***
4. ***Set chart length***
5. ***Dark adapt cells (30 seconds)***
6. ***Start RLC***

Steps ***1 & 2*** optimized signal strength for each sample; minimum fluorescence (F_o) must be sufficiently low (300-500 counts) so that saturating pulses do not yield fluorescence above the instrument measurement capacity of 4000, but must be sufficiently high to record measurable signals. Altering the sensitivity, through changes in PM gain (photomultiplier gain), changed the minimum fluorescence reading. PM gain settings run from 1-30 and are adjusted to each sample type accordingly. The lower the PM gain, the less sensitive the detector becomes; the low gains are used for highly concentrated phytoplankton samples. In contrast, the higher the PM gain, the more sensitive it becomes, allowing for viability detection of more dilute concentrations of phytoplankton, but also increased background noise. It is important to note that the lower

PM gain, the better the likelihood of detecting whether or not a sample was deemed viable since the variable "noise" of the baseline fluorescence was reduced.

Once the PM gain had been adjusted, step ***3*** involved 'blanking the instrument' with purified Milli-Q water itself and optical crosstalk between excitation light and detector response. Milli-Q and filtered sea water gave similar levels of fluorescence, CV <5%, so for the purposes of consistency, Milli-Q was used for blanking purposes. Also, any time the PM gain was changed, the instrument had to be re-zeroed to remove the baseline increase in noise. However at higher PM gains (20+), even the baseline fluorescence of Milli-Q varied after blanking. To be consistent, the instrument was blanked once at those settings for all replicate samples so as not to introduce any additional blanking effects.

For step ***4***, it was important to set the software 'chart length' to the desired time before starting a RLC experiment (nominally 20 minutes), so that the chart did not get cut off mid experiment; the chart length importantly controlled the collection frequency of the stored fluorescence time-series for each sample run (the raw data used in this thesis). Samples were dark-adapted for at least 30 seconds (step ***5***) to allow maximum opening of reaction centers and to ensure the best contrast between minimum and maximum levels of fluorescence. It is key to note that all samples were continuously mixed with a battery powered stir bar to prevent cells from settling and creating an inconsistent signal. Once the chart length was set and the cells were dark-adapted, running a RLC + Recovery (step ***6***) was as simple as pressing a button since it was already preprogrammed in the software.

CONCENTRATION EXPERIMENTS

A few concentration experiments were conducted in order to determine whether concentrating treatment samples (low F_v/F_m values) would increase the signal by increasing the possibility that additional live cells existed in the sample and would be detected by the PAM. Two methods of concentration were used, centrifugation and gravity filtration. To control for potential concentration effects (i.e. loss of cells or cell damage), untreated control water was also concentrated using the same methods (but only with 10 fold concentration). Before and after concentration, water samples were run on the PAM to generate RLC data so that results could be directly compared. It is important to note that concentration and resuspension by either method is not exact, so actual concentrations were later calculated using the change in the PM gain from the complementary sample measurements. Furthermore, whole water samples were used for the purposes of these concentration experiments.

Centrifugation

In order to concentrate through centrifugation, 50mL carboy subsamples were placed into a 50mL Falcon tubes and then spun down at 6000 rpm for a period of 10 minutes.

- **10 fold** (1 replicate/sample): 45mL water was carefully drawn out so the pellet particulate floc stayed in the remaining 5mL. The pellet was resuspended through 8 seconds of vortexing before running a RLC experiment.

- **100 fold** (8 replicates/sample): 45mL drawn out of each replicate and the remaining 5mL (with the pellet) for each replicate was combined into one falcon tube and centrifuged again. The final step was to remove 36mL and leave the pellet in the remaining 4mL, resulting in a 100 fold concentration. The pellet was resuspended with 8 seconds of vortexing before running a RLC experiment (for treated samples only, otherwise the concentrated control water would be too dense for the PAM to read).

Gravity Filtration

To concentrate through gravity fed filtration, an aliquot of sample water was poured into a funnel containing a 10 μm filter. After concentration, the nylon filter was removed and placed in a 15mL Falcon tube with a small volume of filtered water. The particulates were resuspended through vortexing.

- **10 fold:** 100mL of sample water was poured through the filter, placed into 10mL of the leftover filtered water and vortexed for 8 seconds. This was done only for the two control samples in order to get two types of concentration methods.

- **100 fold:** 400mL of sample water was poured through the filter, placed into 4mL of the leftover filtered water and vortexed for 8 seconds. This was done only for the two treated samples as a way to get an increased number of overall concentration samples as well as additional methods for comparison.

Blank Determinations

Both Milli-Q (purified water) and filtered sea water (FSW) samples were also run on a RLC in order to characterize the signal that water itself may contribute to fluorescence. Milli-Q water was obtained from a Milli-Q machine in the Biological Oceanography lab, while the FSW was created through filtration. Filtered seawater was collected from the aquarium room at Moss Landing Marine Laboratories, but to ensure cell removal, the water was pressed through a Millex (.45 μm) filter attached to the nozzle of a 50mL plastic syringe. Three replicates of each water type were used for RLC experiments.

Results

Series of Custom Algorithms

The series of algorithms created using MATLAB were designed to automate file processing, compare methods of rapid light curve (RLC) peak calculations to WALZ WinControl, and to analyze the main RLC trends for biological activity indicators. One issue to consider during the development of the algorithms for analyzing trends was that RLCs generated from natural seawater samples (Figure 6B) were not as defined as RLCs generated from phytoplankton cultures (Figure 6A). Cultures can be grown to dense chlorophyll concentrations on the order of 500-1000 μg/L, while natural seawater sample have on the order of 1-2 μg chlorophyll/L or less. Instrument noise therefore became more of a problem for defining viability because the RLC baseline fluorescence of natural, healthy, control samples was highly variable compared the flat baseline in the healthy cultures; this is essentially the focus of this thesis – evaluating fluorescence signal at the detection limits of the instrumentation. Furthermore, the natural sample maximum fluorescence F_m did not always decrease sequentially with each subsequent saturation pulse. However, it was still possible to extract the features that suggest biological activity, such as the dark-adapted F_v/F_m ratio (Figure 6B, #1) and the baseline fluorescence slope (Figure 6B, #2). Furthermore, other methods were devised to help corroborate those two features. The methods that worked best included: assessment of the quality of the F_v/F_m peak from signal to noise analysis (confirming the validity of the WALZ F_v/F_m measurement) and an overall RLC calculation that resulted in a bulk numerical value to 'bulk' characterize the entire RLC.

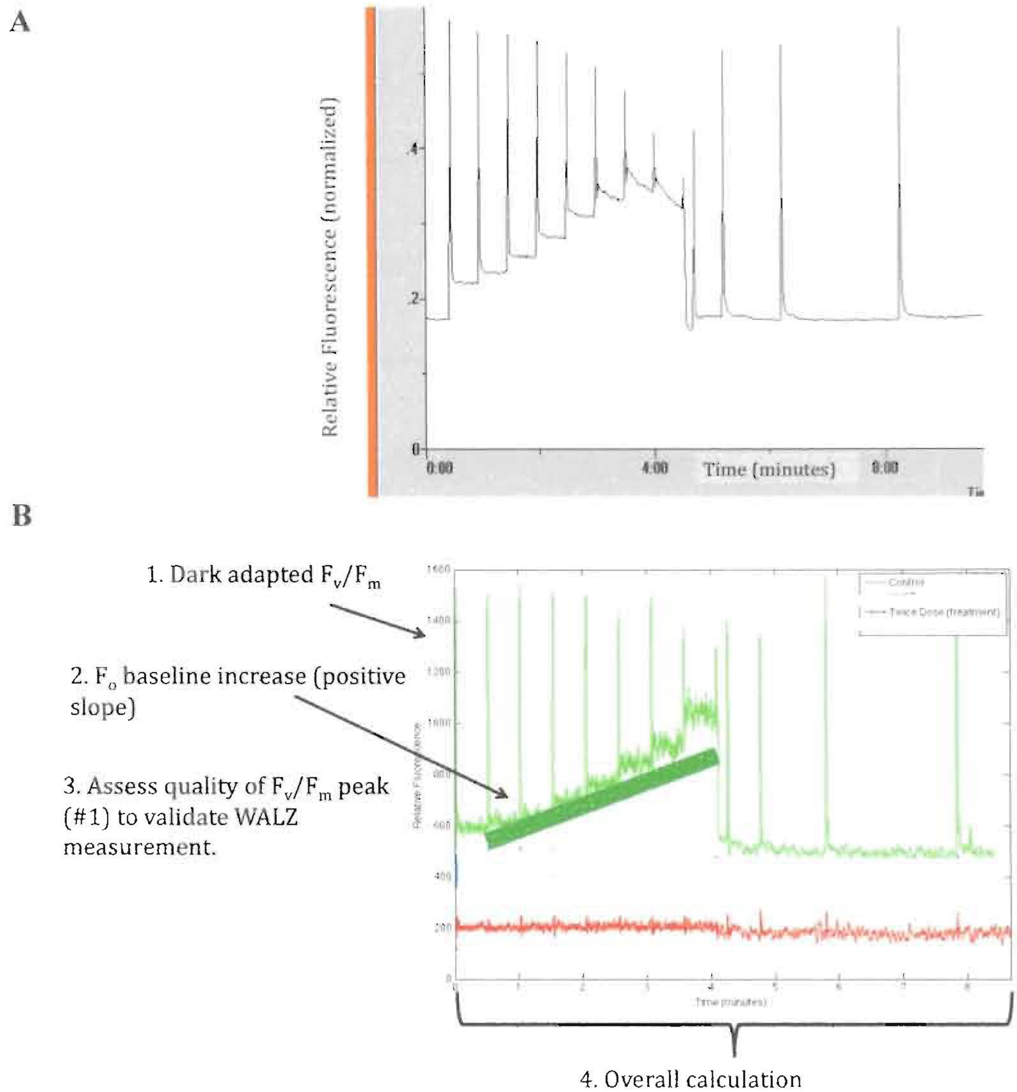

Figure 6. Programmed rapid light curve fluorescence trace showing *Tetraselmis* sp. culture (A) compared to natural seawater samples (B).

The second part of this analytical trend process was to set thresholds for minimum detectable biological activity for each of the four features: the dark-adapted F_v/F_m ratio, the quality (signal to noise ratio) of the dark-adapted F_v/F_m peak, the minimum fluorescence baseline slope value, and the overall 'bulk' characterization parameter. The goal of this effort was to quantify a numerical difference between the control and treatment samples in order to more confidently determine whether or not a biological response was present. The algorithms created for this thesis included algorithms that: 1) organized data, 2) processed the RLCs, 3) quantified trend thresholds and 4) tested an alternative smoothing approach (Figure 7).

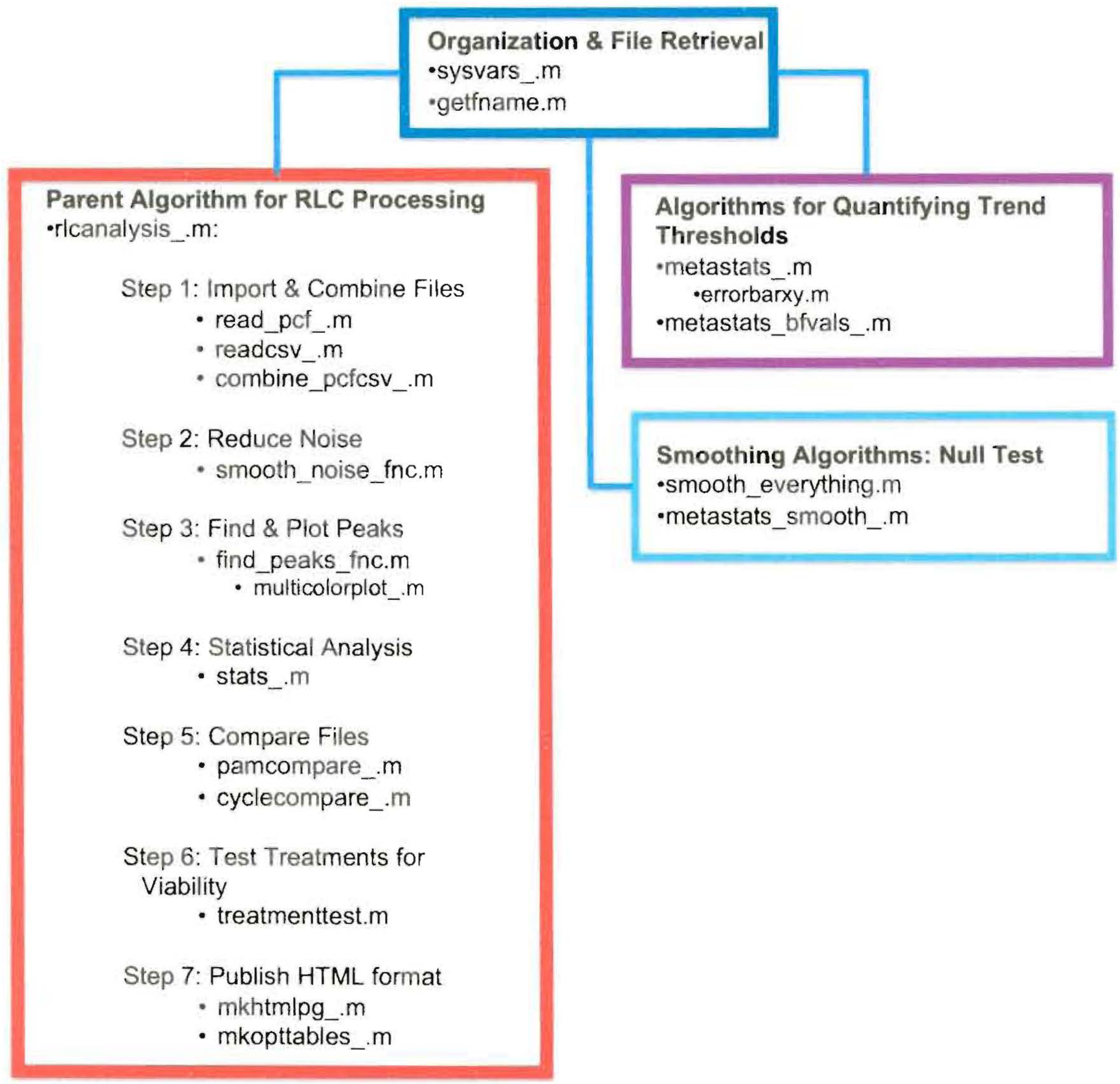

Figure 7. Flow Chart of Algorithms Written Using MATLAB.

The description of each algorithm is described on the following pages. The actual code is in Appendix A, however contact Moss Landing Marine Laboratories Library to access the actual (and updated) code stored in the digital commons.

Organization and File Retrieval Algorithms

sysvars_.m

A script that defined processed files' directories, e.g. imported files and analyzed RLC data (figures, calculations, etc.), so that they were organized and placed into predetermined locations only.

getfname.m

This function (written by B. Schlining) retrieves the name of all specified files in a directory and put them in the form of a character array in the output in order to help the automated importing process.

Rapid Light Curve Data Processing Algorithms

Below is a list of algorithms (with brief description of purpose) that I developed for processing the RLC data; they are included in the seven steps as outlined in the flow chart in Figure 7: 1) importing files, 2) reducing noise, 3) finding peaks, 4) analyzing trends, 5) comparing cycle samples, 6) evaluating treatment sample viability and 7) publishing to an HTML format.

rlcanalysis_.m

This is the parent function I created that runs the entire seven-step program using the functions described below.

read_pcf_.m

This function I wrote was a translation of the WALZ WinControl code (generously sent by the manufacturer) used to create their custom pam chart files (PCF) for Win-Control v2.8. PCFs contain multiple RLC experiment chart fluorescence data, so having a code that would import the PCFs directly into MATLAB was preferred over the process of exporting each RLC from every PCF as a text file (a readable MATLAB format).

readcsv_.m

This function I developed imported the report CSV file containing critical machine setting information and created new columns to store the PM gain used for each

RLC. This information was not stored in the PCF, which made it necessary to save both the PCF and CSV files to use the entire *rlcanalysis_.m* program.

combine_pcfcsv_.m

This function I created combined the imported PCF and CSV files for a specified filename into one matfile for analysis (Figure 8). The output included three variables: *fluorescence* (each column contained the measured fluorescence points), *pamp* (calculated PAM parameters in the report file: F_o, F_m, F_v/F_m etc…), and *settings* (instrument PM/sensitivity gains).

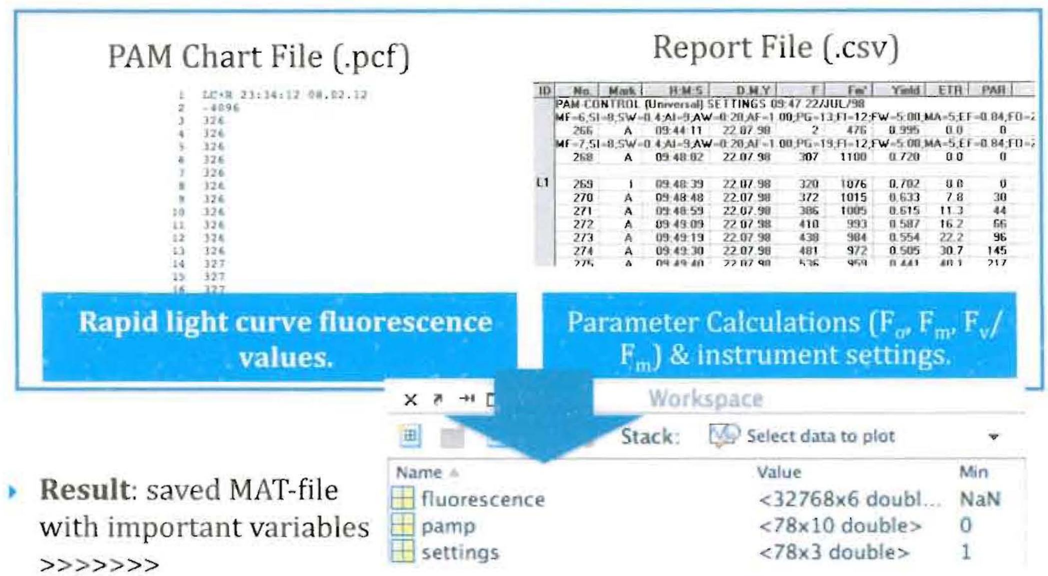

Figure 8. Example of algorithm *combine_pcfcsv_.m* creating variables from imported pcf and csv files in MATLAB.

smooth_noise_fnc.m

Algorithm Development Background

The PAM chart file is a time-series of fluorescence measurements induced by the modulated measuring-light exictation LEDs; the raw chart data used in this thesis were collected at a frequency of 25 data recordings per second. The purpose of optical detectors is to detect peaks or signal during a period of forcing, such as during a light pulse. A signal is a defined as a description of how one parameter is related to another. In the case of peaks or signals during a RLC, the vertical y-axis represents the response of the sample to measuring light excitation as relative fluorescence, while the horizontal x-axis represents time. Methods for digital signal processing include reducing interference and noise to measuring signal positions, height, widths, and calculating the slope changes between points.

In order to increase signal detection, it was important to first reduce noise or interference. One method for eliminating low-level noise is by setting a 'threshold' from optical outputs (Jordanov, Hall and Kastner 2003). However, this type of analysis can create a signal with peaks that are reduced in size, meaning some true peak signal is possibly lost. This phenomenon is more likely to occur when the sensitivity is increased and approaches the instrument's level of detection. Even if a 'threshold' is not used in the optical software, increased sensitivity can lead to an increase in background noise. Both cases present the probability that a real experimental signal will be lost within or within the noise. The advantage of using a RLC experiment is that the WALZ software can be programmed to pulse light at the exact same time for every experiment. This allows for accurate prediction of where peaks *should* appear regardless of the sensitivity level used during the RLC.

The next step in digital processing is to optimize and measure the signal. This is usually done by transforming the data in some way or by applying a filter to generate the best signal. Signals can be simple repetitive waveforms, such as sine or square waves, and the deviations can be described by the peak-to-peak amplitudes. Due to the repetitive waveforms of sine and cosine, Fourier transformations are often used in digital signal processing since they can recover the amplitude of each wave using an integral (Taneja 2008). However, most signals do not display distinctive peak-to-peak values like in cosine or sine waveforms because they are more random in nature. The signals formed from a random process usually possess a bell shaped curve, also known as a normal distribution or Gaussian curve (Smith 1997). In those cases, options include applying a Gaussian filter or Gaussian peak-fitting algorithm for responses that approximate a Gaussian curve.

Neither of these peak optimizing or peak finding techniques (or others that are related) were practical for analyzing RLCs due to the unique shape of each peak; the data did not meet Gaussian model assumptions. From a zoomed out perspective, each peak may resemble a small Gaussian spike (Figure 9A); however the true shape of a PAM saturation peak is closer to that of a plateau (Figure 9B). This shape is caused by the saturation pulse shutting down all the reaction centers for the 0.8 s duration and forcing the fluorescence to maximize.

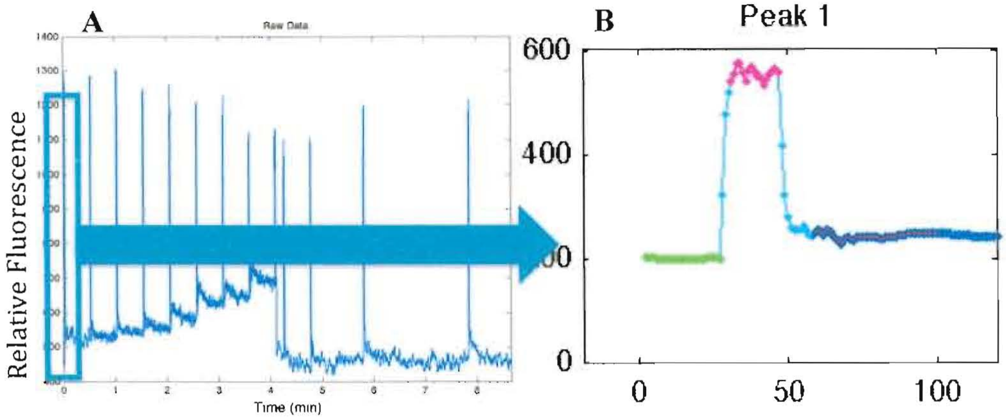

Figure 9. Example of an entire rapid light curve trace (A) and the magnified first peak of that same rapid light curve (B).

Furthermore, applying a Gaussian peak-fitting algorithm requires the assumption, data points approximate a normal distribution, is met. When a few chi-square goodness-of-fit tests (at the 5% significance level) on the RLC chart data in MATLAB were performed, results revealed that neither the entire RLC data set nor the individual peaks followed a normal distribution *(chi-square tests* rejected the null, $p < 0.05$ in both cases). This further reinforced the conclusion that applying a Gaussian peak-fitting algorithm would also not be an appropriate method for peak detection in the unique PAM data series. Overall, a RLC is unlike any typical optical absorbance spectra. It is an experiment with changing forces (e.g. actinic light levels) that test the underlying process of algal fluorescence responses rather than a singular measurement (with one peak) or series of identical replicate measurements. Furthermore, the frequency of light flashes is programmed to change half way through the experiment once it goes into recovery mode. Any peak detection method or overall RLC analysis had to take into account these unique properties to generate the most valid results.

Table 4. Peak bins used in the 20 minute chart length for data processing in the series of algorithms.

Bin	Point Range	Bin	Point Range
1	1-100	8	5300-5450
2	700-850	9	6075-6200
3	1450-1650	10	6300-6450
4	2200-2400	11	7070-7220
5	3000-3200	12	8600-8800
6	3775-3925	13	11650-11850
7	4500-4700		

During the algorithm development process, it was also important to know that the chart data from the PCFs contained only the raw fluorescence values, no accompanying time data. In order to predict where peaks should occur for analysis, it was necessary to calculate the approximate sampling rate based on the chart length. According to the WALZ WinControl v2.8 manual, each chart records a maximum of 30,000 points regardless of chart length selected. Shorter chart lengths therefore had higher resolution since more data points constituted a peak (in a healthy sample). For congruity, all RLCs were run on the chart length setting of 20 minutes with the entire experiment taking approximately 8 minutes (to get 13 total peaks). Knowing that each chart file contained a maximum of 30,000 points meant that for a 20 minute chart file, there were approximately 1,500 points per minute. However, due to the lack of recorded time data, it was necessary to examine multiple charts containing healthy samples with well defined peaks in order to create bins containing specific point ranges where peaks were expected to occur (with some buffer due variations between samples, Table 4). During the light part of the light curve, a SAT pulse was applied every thirty seconds, while the time between the SAT pulses during the recovery portion of the light curve increased between each of the last four peaks. The point range for each peak bin is defined in Table 4 for the 20 minute chart length option. The technique of 'binning' is a solution commonly applied to data sets that have signals too large for random chance (like a RLC peak) and the error of falsely identifying a peak is reduced by breaking up the data into shorter sections and calculating the statistics or measurements for each section individually (Smith 1997). In order to minimize noise and maximize individual peak detection in the following algorithm (*find_peaks_fnc.m*), the *smooth_noise_fnc.m* algorithm was designed to smooth the noise with a moving boxcar average and leave the regions where peaks should occur unaltered.

Algorithm Description

The *smooth_noise_fnc.m* function I created allowed for flexibility on the part of the user by offering a couple options regarding data normalization and chart length input. The first user input allowed the option for choosing whether to normalize the fluorescence data to PM gain 15 (the median sensitivity for the PAM; PM gain values range from 1-30) or to proceed with the smoothing using the raw fluorescence data only. For the purposes of this thesis, all the chart data was normalized to PM 15 to allow comparison between different samples run on different sensitivity levels. Figure 10 below shows examples of RLC traces for each type of sample (control, half-dose, twice-dose (treatment), and blank) before any smoothing or normalizing algorithms were applied.

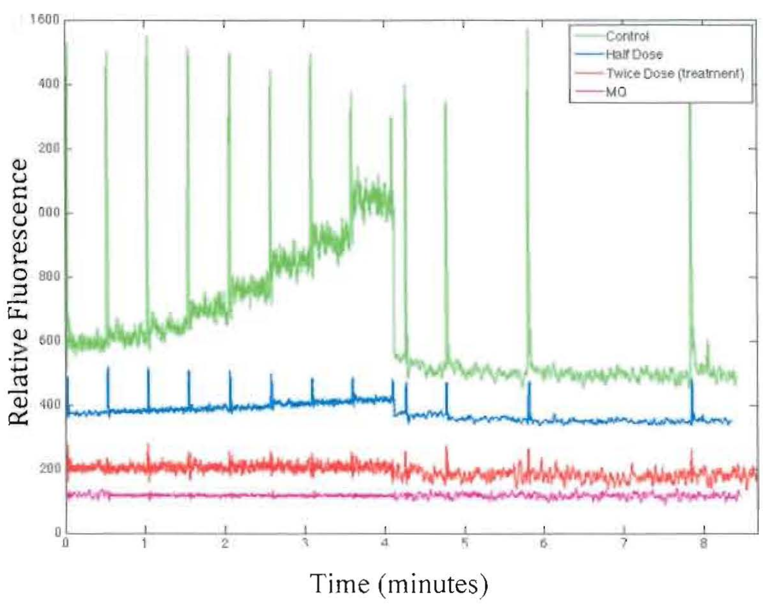

Figure 10. Example rapid light curve fluorescence traces for control, single dose (half-dose), double dose treatment and Milli-Q water samples.

The second user input prompted for chart length (currently only tailored to the options for 10, 15 and 20 minutes), so the proper boxcar smoothing function could be applied to the data. Once the options were logged, the data noise for every chart in the file was reduced through smoothing. During the smoothing process, the peaks were temporarily removed (based on predetermined bins), so the baseline fluorescence noise was averaged and therefore reduced. After, the peak data (either raw or normalized) was overlaid onto the smoothed data and saved in a new file with a special extension ('_s' for raw smoothed or '_ns' normalized smoothed) for further peak analysis. Example output for this algorithm is shown below in Figure 11. The control RLC before the normalizing and smoothing algorithm is shown in Figure 11A, and the same control sample after being processed is shown in Figure 11B. Figure 11A and 11B look very similar, except Figure 11B has a smaller range of relative fluorescence values on the y-axis. Normalizing only reduced the magnitude of the data values since everything was multiplied or divided by a constant; it did not change the trends or peaks that were analyzed later in the process. The smoothing portion of the algorithm averaged the points between the peaks, so the control RLC in Figure 11 B shows less variability in the baseline fluorescence. Figure 11 C shows the treatment sample RLC before the normalizing and smoothing algorithm and Figure 11 D shows that same treatment RCL after algorithm processing. While the treatment RLC peaks in Figure 11 C are not very prominent compared to the noise, once the data is processed by algorithm and the noise is reduced, the possibility of peaks begin to emerge (Figure 11 D).

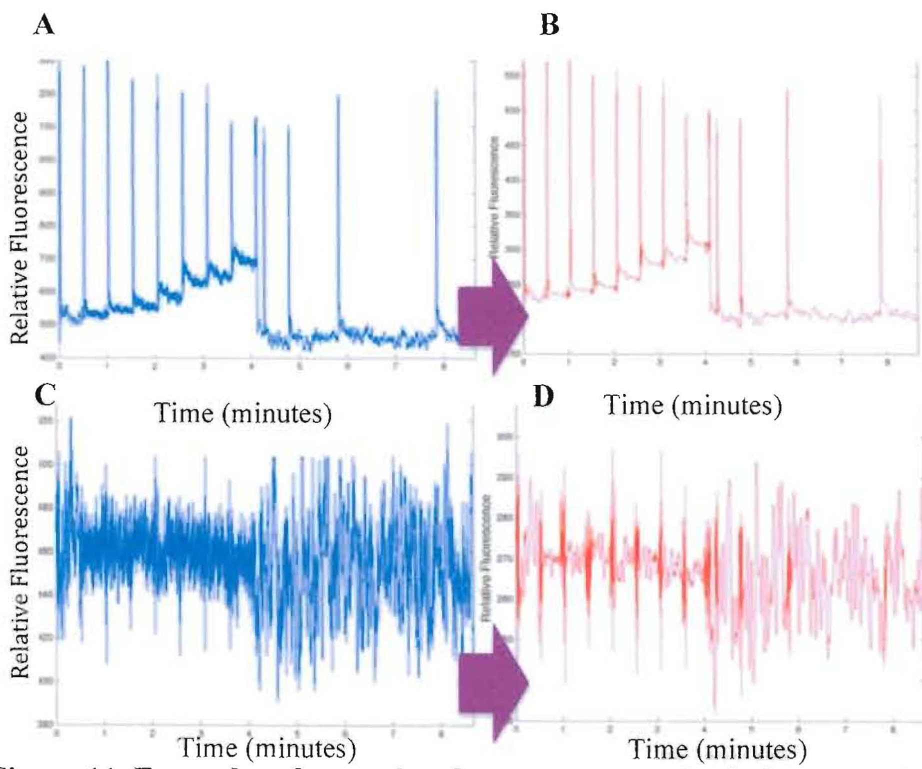

Figure 11. Examples of control and treatment samples before and after smoothing/normalizing algorithm.

find_peaks_fnc.m

This algorithm I developed used a 'peak finder' common to other data sets (chromatograms, spectral analysis); the algorithm moved along the time-series data points to find the changes in slope (change in fluorescence divided by the change in time) within the predetermined peak bins. If the slope fell above a specified threshold, the next steps were to find the maxima and the minima on both sides of the peak. This was accomplished by locating the left hand minima or peak start (green points, Figure 12 A), then following the slope (left side turquoise points, Figure 12 A) up to the local maxima (pink points on plateau shaped peak, Figure 12 A) and then following the second slope down to the right hand side of the peak end to the other local minima (right side turquoise points, Figure 12 A).

The only concerns associated with using any sort of peak calculating method for the RLC included the sensitivity level of the instrument and the quality of the peak. By increasing the sensitivity to detect cell viability in lower cell concentrations or damaged cell samples, there was more variability in the baseline fluorescence. This fluorescence baseline variability led to further uncertainty in quantifying a true 'peak' signal within the noise. Secondly, the ideal 'peak' was not always achieved (Figure 12 A) especially in less than healthy cells. This made the true F_v/F_m ratio of the peak more difficult to determine (Figure 12 B).

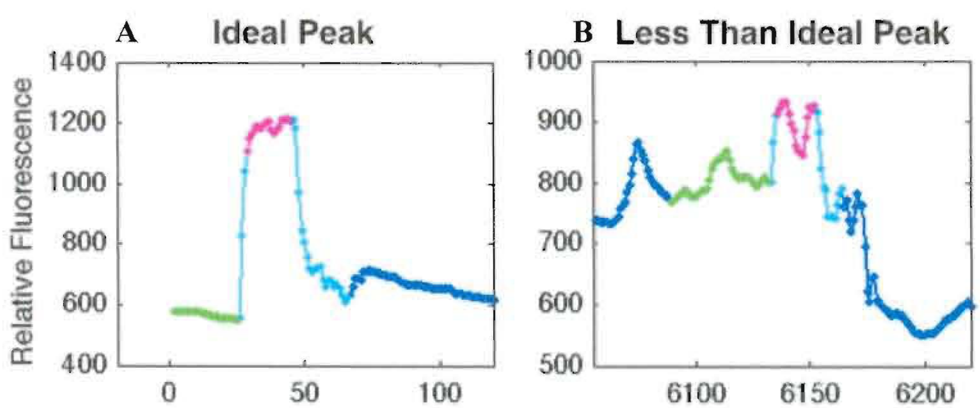

Figure 12. Ideal rapid light curve peak (A) compared to less than ideal peak (B).

Each peak underwent parameter calculations (F_o, F_m, F_v/F_m) using two methods to determine which might be superior to or comparable to WALZ's parameter calculations: single point or a series of averaged points. Therefore, every parameter had three values for comparison. For example: the first F_o calculation was defined as the point at the base of the peak (Figure 13, F_o (1)), the second as the average baseline before the peak (approx. 50 green points, Figure 13, F_o (2)) and the third value was extracted from the WALZ report file. Similarly, the first calculation for F_m was determined as the maximum point of the peak top (Figure 13, F_m (1)) and the second calculation was the average of the points constituting the peak top (approximately 20 pink points, Figure 13, F_m (2)).

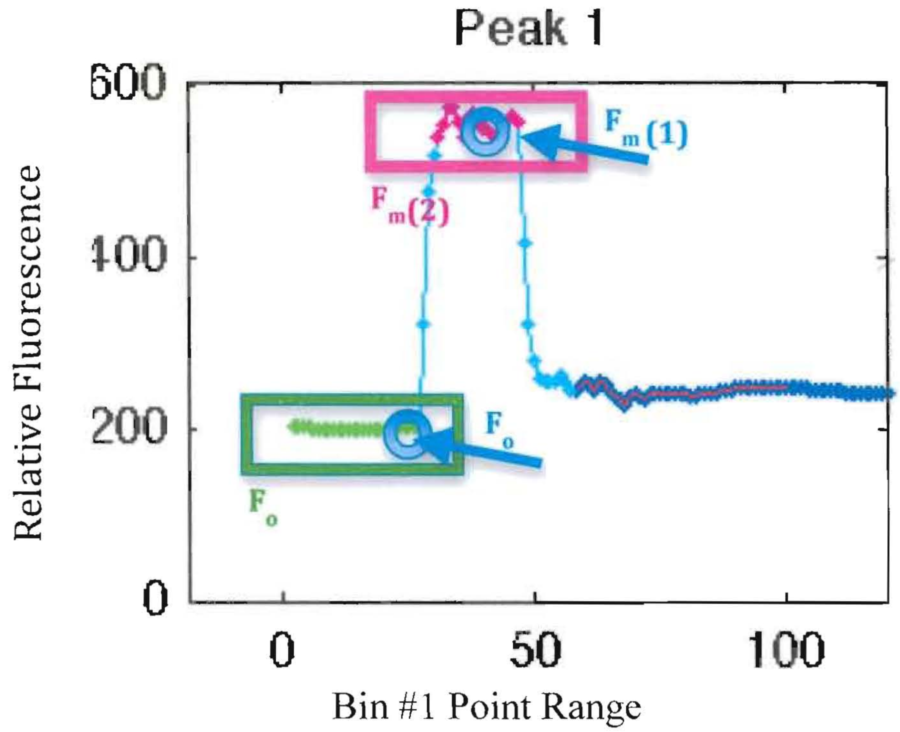

Figure 13. Example of peak and corresponding parameter measurements using the *find_peaks_fnc.m* algorithm.

These basic parameters, F_o and F_m, were used to compute F_v and F_v/F_m in three ways as well (Table 5): F_v using the single point, the average and the WALZ values for F_o and F_m. Similarly, F_v/F_m was calculated using the single point, the average and the WALZ values calculated for F_v and F_m. These fluorescence values for each calculation type (for every peak) are also logged in a table in the output of this program.

Table 5. Parameter calculation methods in the *find_peaks_fnc.m* algorithm.

Parameter	Calculation Methods
F_o	1. One point at base of peak (algorithm) 2. Average of points at base of peak (algorithm) 3. WALZ WinControl
F_m	1. One point at top of peak (algorithm) 2. Average of points at top of peak (algorithm) 3. WALZ WinControl
F_v	1. F_m (1) – F_o (1) 2. F_m (2) – F_o (2) 3. WALZ WinControl
F_v/F_m	1. F_v (1) / F_m (1) 2. F_v (2) / F_m (2) 3. WALZ WinControl

multicolorplot_.m

This function I developed is embedded within the *find_peaks_fnc.m* algorithm so that plots of each rapid light curve could be generated in different colors; blue delineated 'dark' periods (no actinic light) and red indicated the presence of actinic light (Figure 14).

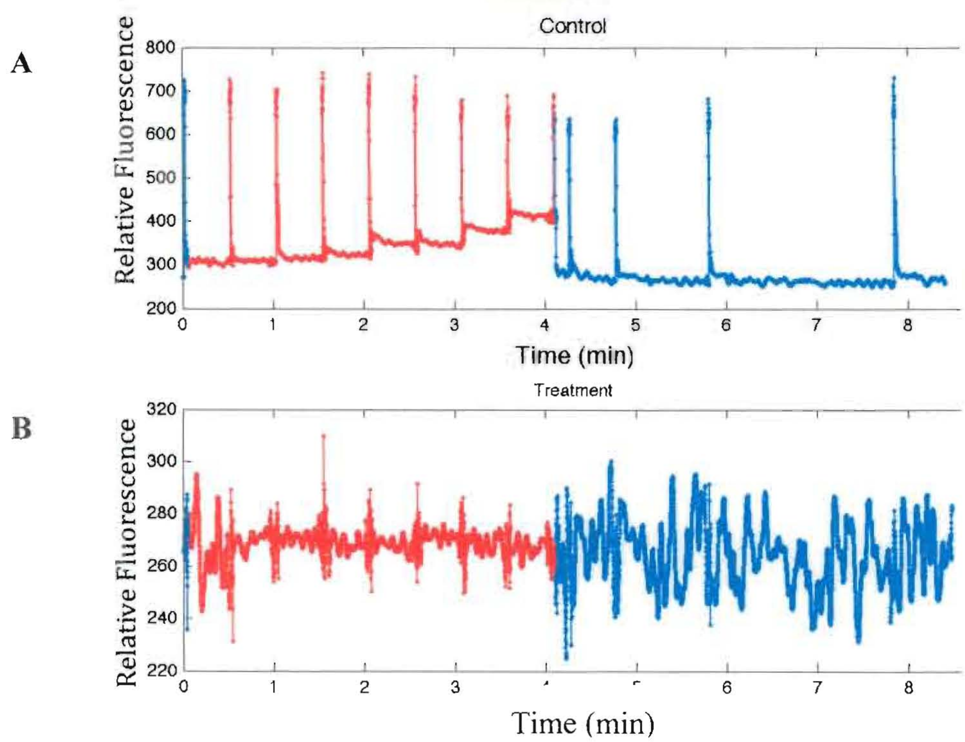

Figure 14. Examples of a control (A) RLC plot and a treatment (B) RLC plot created by the *multicolorplot.m* algorithm.

stats_.m

This algorithm I created calculated 1) the slope of the best fit line through the baseline fluorescence (raw not smoothed), 2) individual peak signal to noise, 3) an overall rapid light curve signal to noise ratio (SNR) and 4) a general 'bulk' characterization value.

A **B**

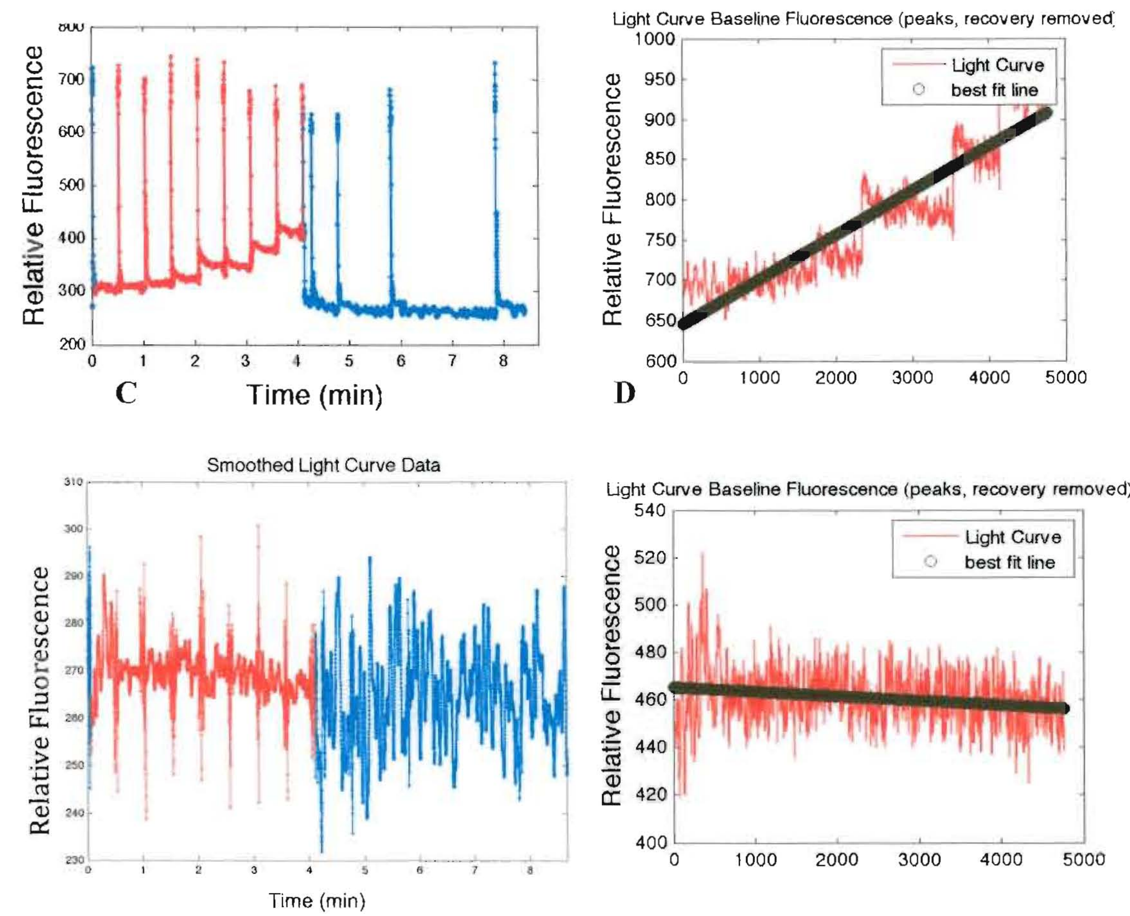

Figure 15. Examples of control and treatment samples of the best-fit slope line calculated for the actinic portion of the rapid light curve.

A general trend for healthy samples during a rapid light curve (RLC) is for the baseline fluorescence to increase as the actinic light increases. For the baseline slope calculations, the recovery portion of the light curve was removed as well as all the peaks from the raw data in order to assess the baseline fluorescence (Figure 15). A best-fit line was computed for the baseline fluorescence and the corresponding slope and intercept were recorded for each sample (half dose samples were not included due to high variability in rapid light curve responses). Figure 15 A shows an example of a control RLC and 15 B shows the best-fit line or slope for the control through the baseline fluorescence during the actinic portion of the RLC (red). Figure 15 C shows an example of a treatment RLC and 15 D shows the best-fit line through the treatment baseline fluorescence during the actinic portion of the RLC (red).

The traditional method for numerically determining whether the data has higher 'signal power' than 'noise', is to calculate the signal to noise ratio (SNR). The SNR equation typically used for population statistics is μ/σ, the ratio of the signal mean (μ) to the standard deviation of the noise (σ). To compute the individual peak SNR mean, each peak top was averaged and divided by the standard deviation of the baseline noise (Figure 16). The mean baseline fluorescence (or noise) technically fluctuates depending on what part of the RLC the mean is taken. Taking the standard deviation of the baseline across the entire RLC (rather than by each segment) is a way to standardize it across all the samples. This would reveal whether a particular peak is detectable above the baseline noise of the RLC.

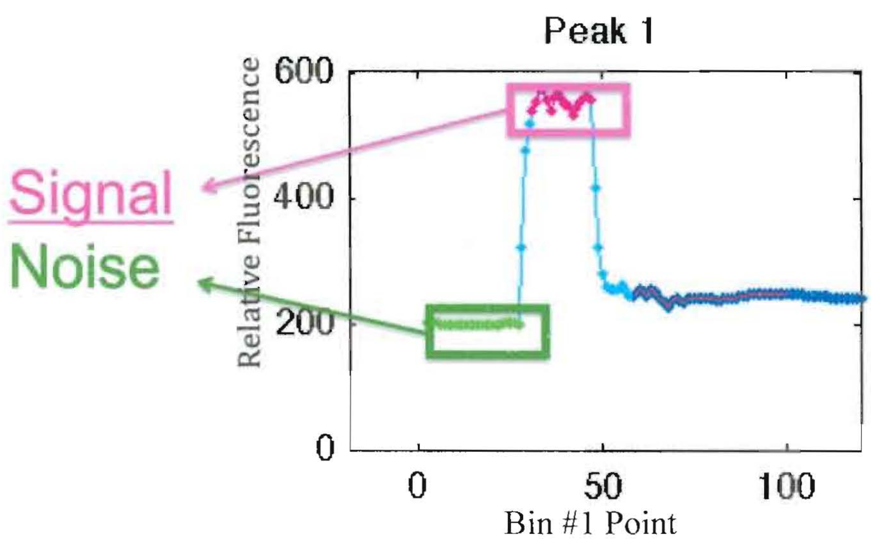

Figure 16. Illustration of SNR peak calculation for *stats_.m* algorithm.

To determine whether a single calculation could assess the RLC chart as a whole and generate a numerical value that represented the overall 'health' of the sample (rather than examining individual aspects like F_v/F_m or baseline slope), two methods were compared: calculation #1 and calculation #2 (discussed in depth below).

Calculation #1 was similar to the method for individual peak SNR calculations, but instead applied to the entire RLC. The signal mean was calculated by taking the average of all the peak tops (Figure 17, pink) in the RLC and dividing the mean by the standard deviation of the baseline noise for that RLC (Figure 17, green). Figure 17 illustrates how this concept was applied to different samples, such as a control (Figure 17 A) and treatment (Figure 17 B) RLC.

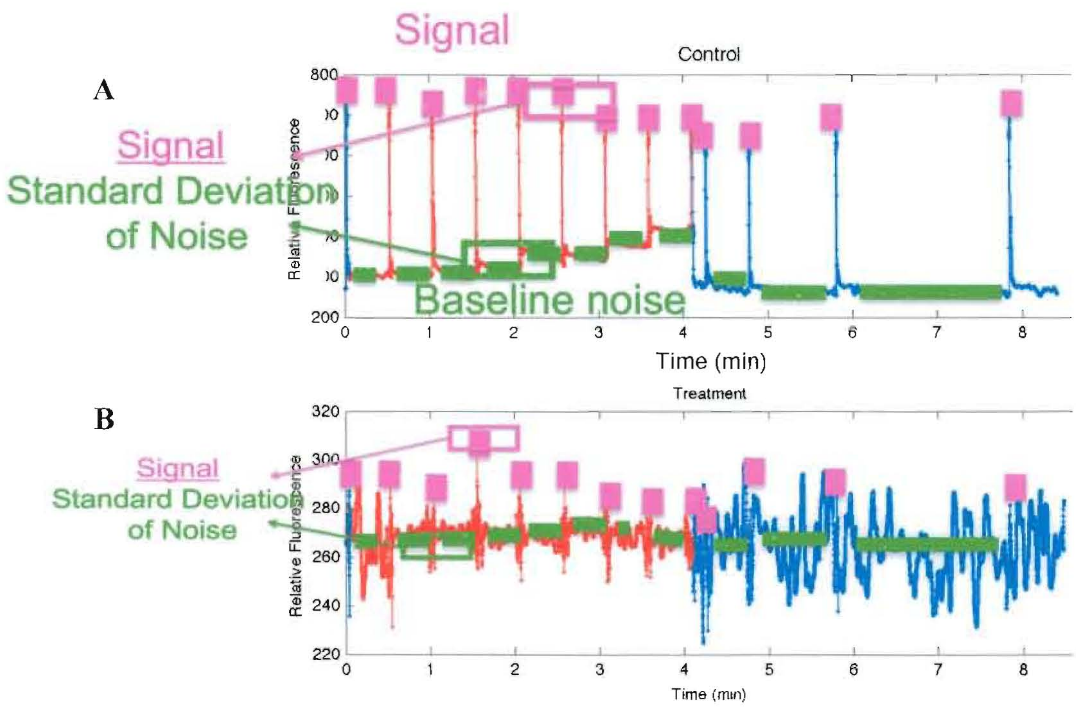

Figure 17. Illustration of method for Calculation #1 designed to numerically characterize the RLC trace for samples (control A, treatment B).

Calculation #2 was a method developed as a second method to 'bulk' characterize the entire RLC using the steps below:

1. ***Normalize data (divide by mean)***
2. ***Take standard deviation of normalized data***
3. ***Find Fm max of normalized data***
4. ***(Standard deviation/F_m max)*10***

Step 1 for data normalization meant finding the mean of the entire RLC and dividing each data point by that calculated mean. This reduced the magnitude of every measurement to hover near zero, allowing direct comparison of control and treatment samples (Figure 18).

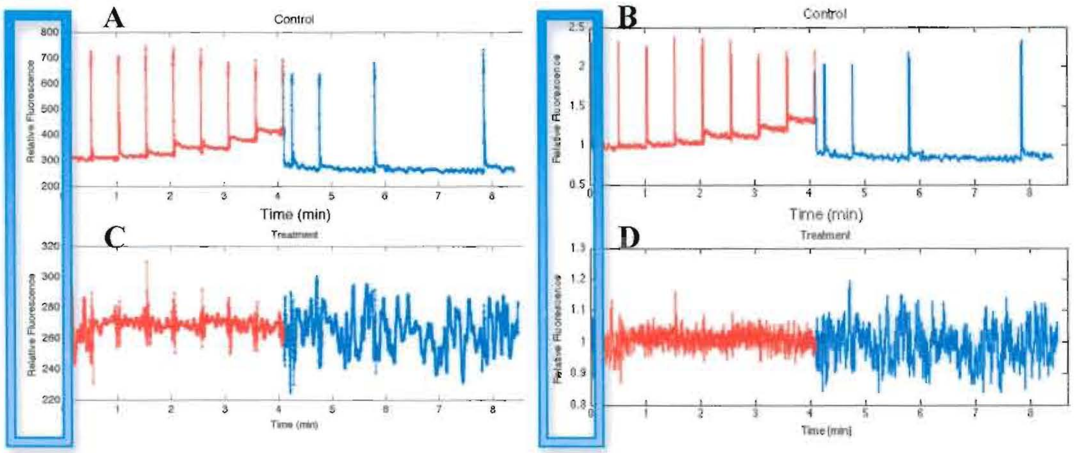

Figure 18. Examples of before and after step #1 for calculation #2 method.

Step 2 computed taking the standard deviation of the entire normalized RLC data. By doing so, it incorporated the rising baseline fluorescence, which showed higher standard deviations in healthy samples (0.1-0.4) compared to treated or non chlorophyll containing samples (0-0.1, Figure 19). This indicated that the underlying process of baseline response to actinic light did affect the standard deviation of the chart since treated or samples that contained no chlorophyll (FSW and MQ) fell at or below 0.1.

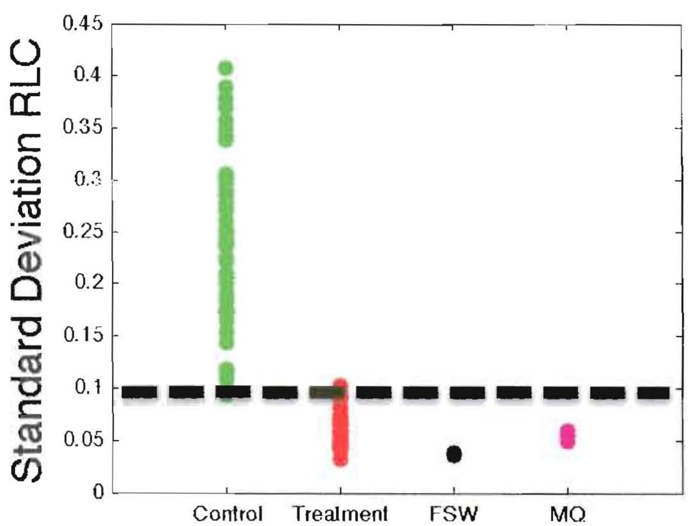

Figure 19. RLC standard deviation calculation results for each sample type.

The third step in the score calculation was to find the maximum point in the normalized data set. Healthier cells exhibited larger fluorescence signals and maximum F_m values (between 1.7-3.7) than treated samples (between 1-2) as shown by figure 20.

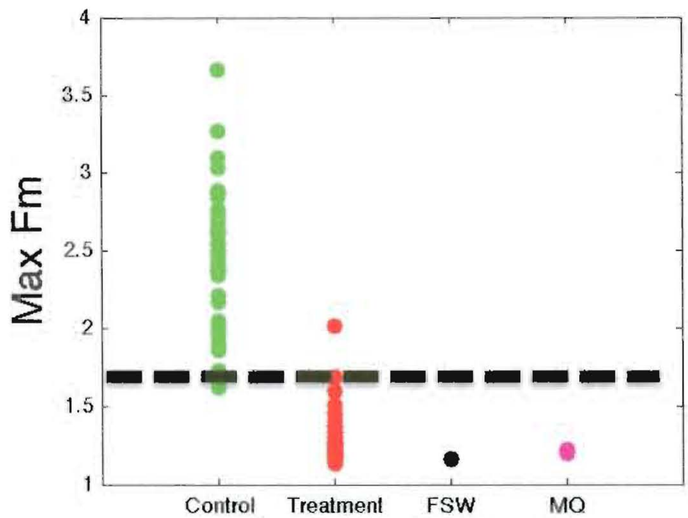

Figure 20. Maximum fluorescence (F_m) calculation results for each sample type.

The fourth step involved taking the standard deviation of the normalized data, dividing it by the maximum value and multiplying by a constant, 10, to get a range of final values between 0-1. This last step is similar to calculating coefficient of variation, but instead of dividing by the mean, the standard deviation is divided by that maximum value.

pamcompare_.m

This function plots a supplemental output of F_v/F_m over the course of each individual light curve (also logged in table format in *find_peaks_fnc.m*) for all three methods for visual comparison: WALZ and both methods computed from the series of algorithms.

cyclecompare_.m

This function compared files side by side as chosen by the user, such as rapid light curves for a control and corresponding treatment. The output included both the plots of the entire normalized smoothed RLCs and for each individual peak within those RLCs compared (Figure 21). The individual peak comparison plots in Figure 21 C and D show a gray bar that outlines the area where a peak is expected to occur within a bin. This allows for visual analysis of whether or not a peak exists in the treated output. Furthermore, this function defined the criteria that must be met in order for the RLC to be considered a viable sample (*see Results section "Defining Thresholds for Quantifying 'No Biological Response'"*).

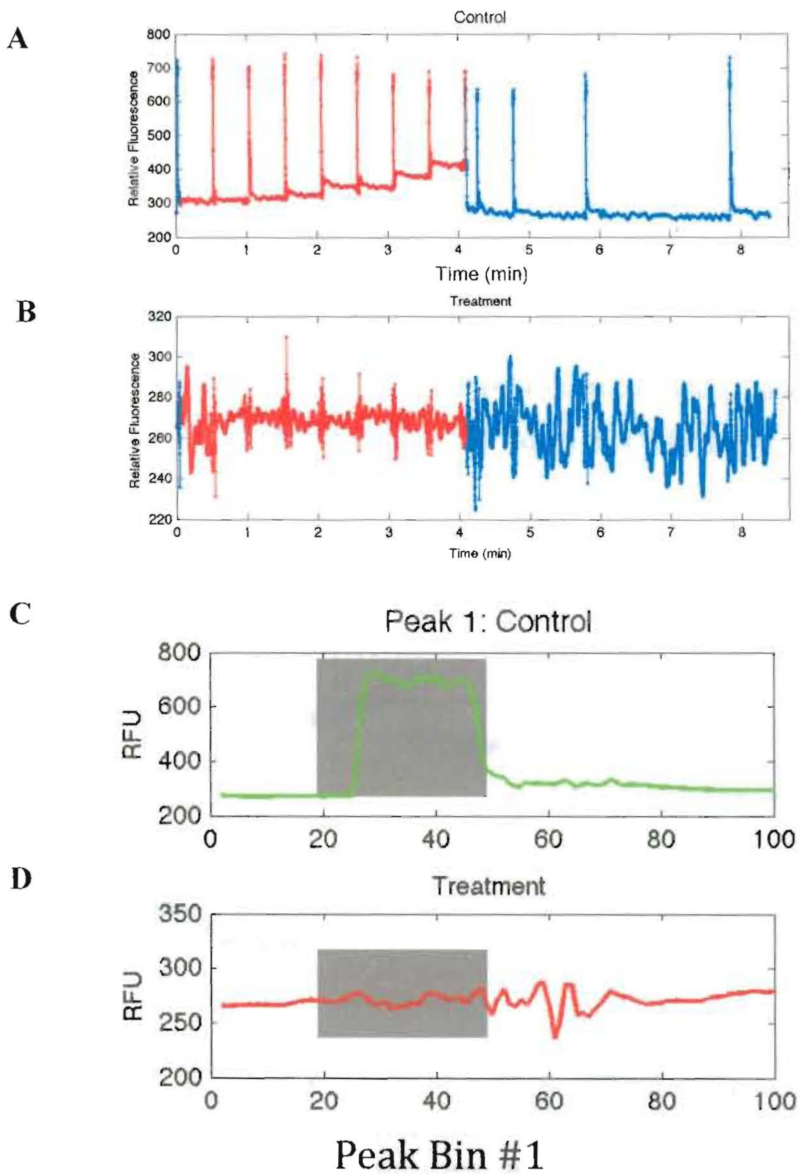

Figure 21. Example of output from *cyclecompare_.m* comparing a control (A) and treatment (B) RLC and peak 1 from the same control (C) and treatment (D) RLC.

mkopttables_.m

This algorithm was designed to make a directory listing all the files that have been made into an html format (*see mkhtmlpg_.m*), to provide a convenient access location for files that have been fully processed.

RAPID LIGHT CURVE STATISTICAL ANALYSIS ALGORITHMS

Below is a list of functions (with brief description of purpose) that I developed for larger scale statistical analysis across experiments.

metastats_.m

This function I designed collects the initial dark-adapted peak data for the series of custom algorithms (the F_v/F_m values and peak quality calculations) and also for the data stored in the standard WALZ report. The next algorithm step was to find the mean value for each of the three 'data quality' criteria and plot the values with standard error bars. The frequencies of the observed values (by sample type) were plotted in a histogram to validate or refute the method or approach. This was the empirical method used to determine a threshold that would separate values generated from the control and treatment samples. These figures are shown later in the Results section. For statistical purposes, a MATLAB function *ttest* was used to conduct paired or two-sample t-tests to test the means for significant differences between each sample type.

The stored, dark-adapted peak data, generated from the custom algorithms of this study (F_o, F_m, F_v, F_v/F_m) were used for direct comparison to identical parameters from the undisclosed computation methods of WALZ. This served as an effective 'quality control' for derived values obtained from custom algorithms relative to standard WALZ methods. The mean initial F_v/F_m ratios between the series of algorithms and WALZ within each sample type were not significant different (Figure 23, *pairwise t-tests*, all p-values > 0.05). The data does also reaffirm the assumption that healthy cells have a higher ratio than stressed cells (Figure 23) since the mean initial F_v/F_m ratios differ between the control and treatment samples. There was a small difference of mean F_v/F_m values between the series of algorithms and WALZ for treatment samples, but it is likely due to the variability of the baseline noise interference (*pairwise t-tests*, all p-values < 0.05).

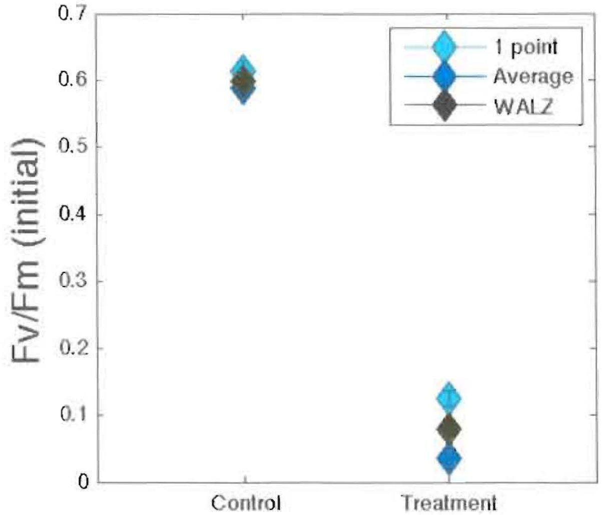

Figure 23. Comparison of initial dark-adapted F_v/F_m calculation between the series of algorithms methods and WALZ.

metastats_bfvals_.m

This function I created collected the baseline slope calculations from all rapid light curve data files and saved the data in a matrix for additional analysis. The next step involved plotting the mean baseline slope for each type of sample as well as a creating a histogram of the frequencies of the observed slope values for establishing a viability threshold. Again, the MATLAB function *ttest* was used to test the means for significant differences between each sample type.

Alternative Smoothing Algorithm

Below is a list of functions (with brief description of purpose) that I modified from my original program or created separately to smooth the raw chart with different methods.

smooth_everything.m

This algorithm I designed is a combination of my original functions *smooth_noise_fnc.m* and *find_peaks_fnc.m*. Instead of removing peaks to smooth the noise, the same boxcar average is applied to the entire chart, peaks and all. Then the new file runs through the rest of the peak finding program (exact same code, just copied and pasted) in order to generate new PAM parameter calculations to be compared to the other PAM parameter calculations method (smoothing only). The purpose of this function was to test the null hypothesis that averaging the whole data set would have no impact on the F_v/F_m signal. This was to ensure that this series of algorithms was indeed a better method for PAM RLC peak detection since it preserved the raw peaks and only averaged the baseline fluorescence. The results of this method showed unaltered control data mean F_v/F_m was about 0.55, but dropped nearly in half to 0.3 after smoothing the entire chart (Figure 24). There were significant differences between the means for both types of data (*pairwise t-tests,* p-values < 0.05), however it is more important to note that the change in control, 'healthy' data indicates a loss of signal using this method in the form of one crucial measurement, Fm. The averaging function lowered all of the values that constituted the top of the peak. Furthermore, the smoothing algorithm wiped most of the first peak (since it occurs within the first few points), making an initial F_v/F_m calculation nearly impossible. For all comparisons in Figure 24, peak 2 was used since it remained intact. Results suggest that due to loss of signal, this is not an ideal approach for analyzing the rapid light curve data and trends; the null hypothesis was rejected.

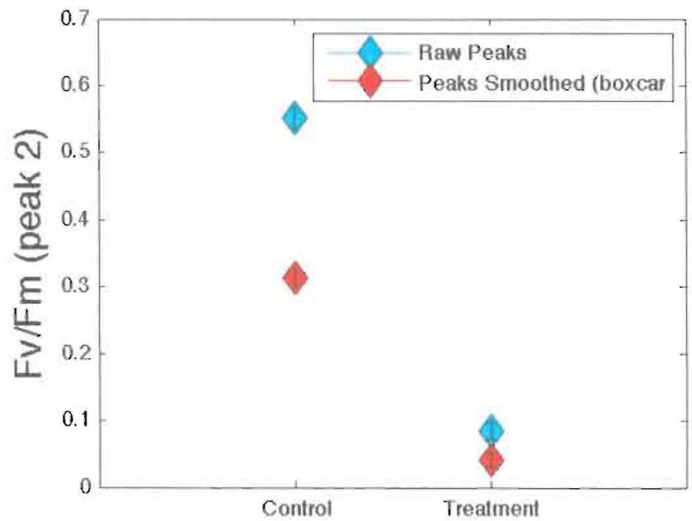

Figure 24. Mean F_v/F_m before and after the *smooth_everything.m* algorithm.

metastats_smooth_.m

This function I developed collected the entire light curve data file with all calculated parameters for the *smooth_everything.m* generated files. Two different experiments (each with 6 uptake and 6 treatment files) were used for comparison of the *smooth_everything.m* program to my series of algorithms parameter calculations.

Defining Thresholds For Quantifying 'No Biological Response'

Dark-Adapted F_v/F_m Ratio

The parameter, F_v/F_m, is commonly used in the assessment of physiological status since this dimensionless ratio has fixed upper and lower limits (~0-0.8) and can be compared among results generated by fluorescence instrumentation from different manufacturers. As a highly sensitive measurement of photosynthetic performance, healthy cells are expected to have F_v/F_m ratios between 0.5-0.8 (green box in Figure 25). However, stress will drive the value down with dead or severely stressed cells often characterized by F_v/F_m values of 0.2 and below (red box in Figure 25). The samples chosen for the this thesis represent a wide range of F_v/F_m values and most of the samples fall within their expected range; the controls have high F_v/F_m ratios (0.5 and above) indicating that they are healthy, while 96% of the treated samples fall below the 0.2 F_v/F_m ratio threshold.

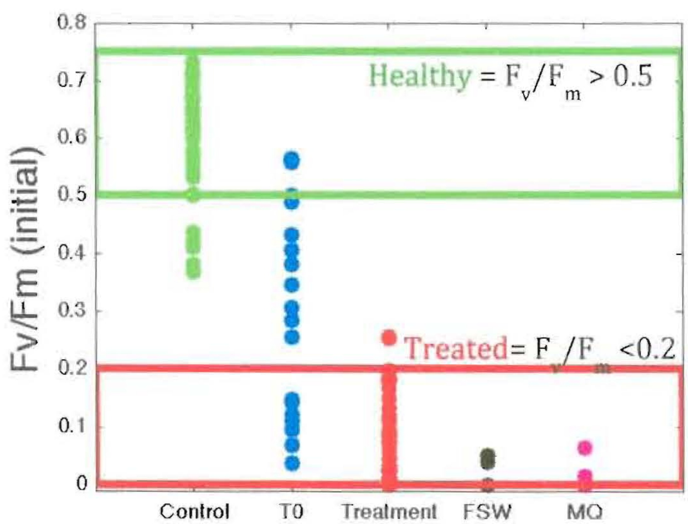

Figure 25. The dark-adapted F_v/F_m ratio categorized by sample type.

Assessing Peak Quality

Defining peak threshold criteria is essential for determining whether a peak exists within the predetermined bins and whether the corresponding F_v/F_m ratio is an adequate measure of the signal. The threshold criteria test involved calculating the signal to noise ratio (SNR) for each peak within a sample type. For statistical analysis, the SNR mean for each sample type was compared to the other sample type SNRs (within a peak number) to determine whether a significant difference existed. While there were significant differences between the mean SNRs of the control and treatment samples for a couple of the initial and final peaks (*pairwise t-test*, p-value <0.05), the standard deviation lines overlapped, suggesting that individual SNR may not be the most clear cut criterion for peak detection. The trend of decreasing mean SNRs for the controls as the RLC progressed during the actinic portion of the light curve, seen in Figure 25, is expected since the peaks get progressively smaller due to non-photochemical quenching.

Even though the results did not show clear delineations to be considered a criterion for viability, this SNR method was investigated further to determine if it would still be a good method to assess the quality of the dark-adapted peak (criterion #1) since the means between the control and treatment differ by an order of magnitude. In the histogram on Figure 26 B, 90% of the control SNR values fall above 50 and 94% of the treatment below 50. This suggests that the healthier cell samples tend to have higher SNR than treated samples, which is also reflected in the corresponding F_v/F_m ratios. Using this data, a peak quality threshold was established at a SNR of 50; below this value indicates poor quality and above indicates a peak distinguishable from the baseline noise.

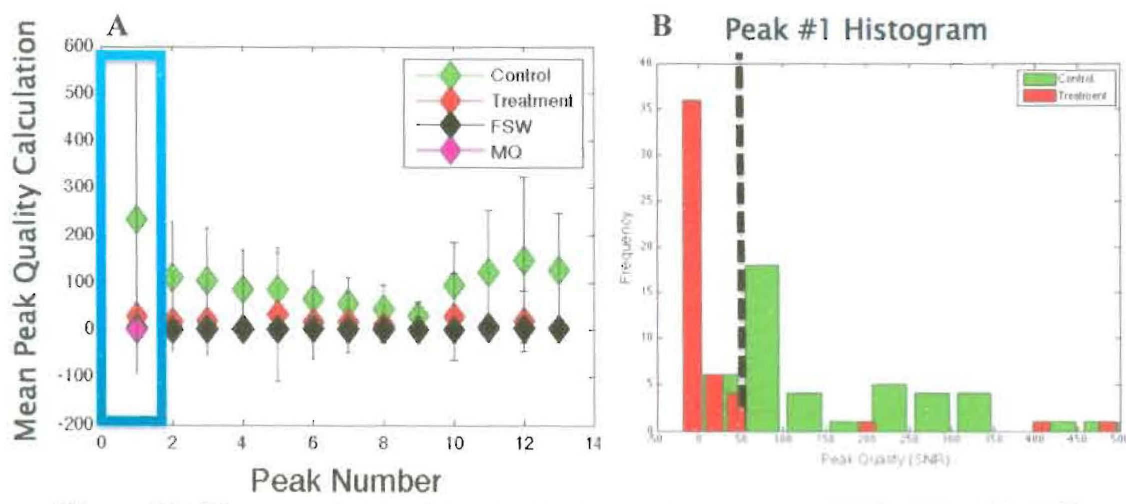

Figure 26. Mean peak signal to noise ratio according to sample type (A) and frequency of observed ratios (B) for the dark adapted peak #1.

Baseline Slope of F_o in Presence of Actinic Light

A general trend for healthy samples during a rapid light curve (RLC) is for the baseline fluorescence, F_o, to increase as the actinic light increases. A best-fit line or was computed for the baseline fluorescence and the corresponding slope and intercept were averaged for each sample type. The observed means of the baseline slope suggest that there are significant differences between the control water and the rest of the samples (Figure 27, *pairwise with control; two sample t-tests with MQ and FSW*, all p-values < 0.05). The average slope for the control samples was 0.054 (+/- 0.0043 STE), while the other three mean slopes were negative or near zero. Treatment, Milli-Q and FSW samples were not significantly different from one another (*two sample t-test*, p-values > 0.05). This suggests that healthier cells tend to have a larger positive slope during the actinic light period of the RLC, while stressed cells or cell free water samples tend to have slopes closer to zero.

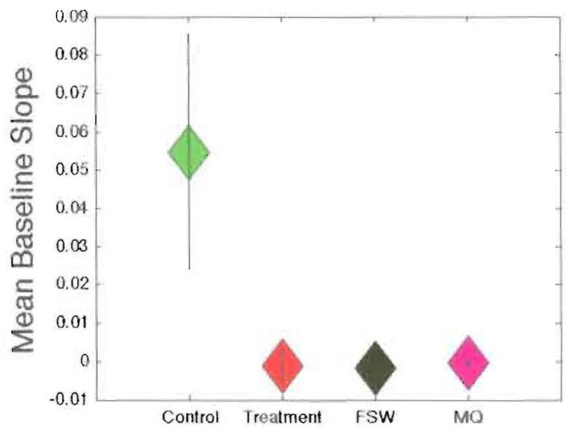

Figure 27. Mean baseline slope according to sample type.

To determine a threshold for using the baseline slope as a criterion for viability, a histogram (Figure 28) plotting the frequencies of the observed slope values for the control and treatment samples is shown in this histogram. The black dashed line in Figure 28 highlights the delineation between control and treatment samples; 94% of the calculated control baseline slope values (in green) fall above a slope of 0.02 and 100% of the treatments (in red) below that threshold. Using this data, a baseline slope threshold was established at 0.02; below indicates lack of response to actinic light and above indicates viability from response to the actinic light.

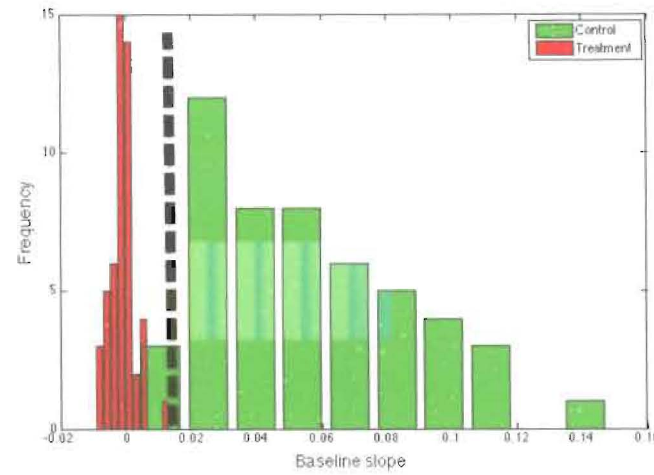

Figure 28. Histogram of baseline slope values for control and treatment samples.

'Bulk' Rapid Light Curve Characterization

To assess each RLC in its entirety, rather than just the initial F_v/F_m peak, two methods were assessed to determine which would yield a better numerical indicator for viability and quantifying the absence of biological activity.

Calculation #1

The results of Calculation #1 for bulk or gross characterization of entire rapid light curve show that there is high variability in the calculated mean values for the controls, seen Figure 29 below. Even though treatment, FSW, MQ and showed small standard deviation bars, the wide overlap of results for the controls reinforced that it is not a good gross approximation of the rapid light curve trends. This is further reflected by the histogram showing the frequency of calculated values for the control and treatment samples on Figure 30; if this calculation #1 method were a good 'bulk' or gross approximation, then would show two distinct populations. However, there exists high overlap of the red treatment and green control bars between 0 and 50, even though the control values range from zero to above 400. Below that threshold of 50, 100% of the treatments lie there, but so do approximately 40% of the controls. Due to these results, this calculation method was not used to establish a fourth criterion of viability.

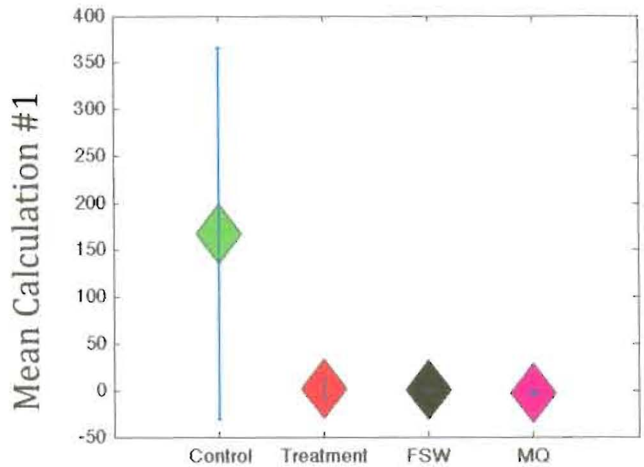

Figure 29. Mean calculation #1 values across sample types.

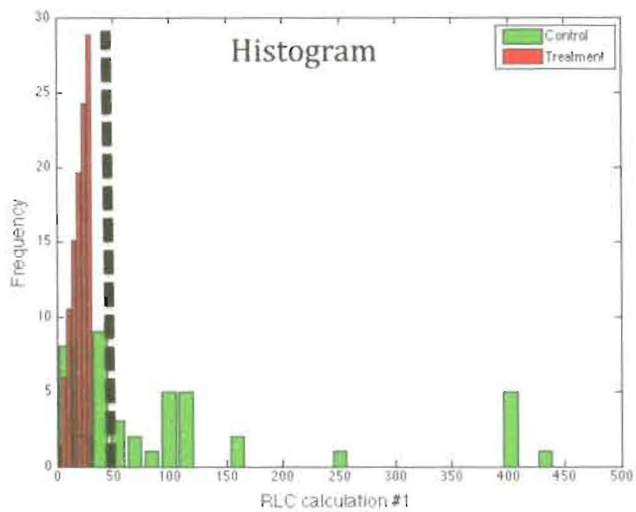

Figure 30. Histogram of RLC calculation #1 values for control and treatment samples.

Calculation #2

The mean results of Calculation #2 for bulk or gross characterization of entire rapid light curve are shown on Figure 31. There is some variability in the calculated mean values for the controls (long standard deviation bars), but the control mean is significantly different from treatment (*pairwise t-test,* p-value < 0.05), FSW, and MQ samples (*two sample t-tests,* p-values < 0.05 for each blank sample type). It is apparent even from the mean plot that there is a separation between control and low biological response samples. The average score of a healthy control sample is around 1, while treatment and water samples fall below 0.65 (Figure 31). This separation is further reinforced by the histogram (Figure 32) showing the frequency of calculated values for the control and treatment samples. Figure 32 illustrates little overlap between the calculation #2 values for the control and the treatment samples; 96% of the control samples having values above 0.65 and 92% of the treatments and all of the blank water samples below 0.65. This suggests that using calculation #2 would be useful for confirming cell sample viability. Therefore, a calculation #2 criterion value threshold was established at 0.65; below indicates lack of biological response above indicates cell viability.

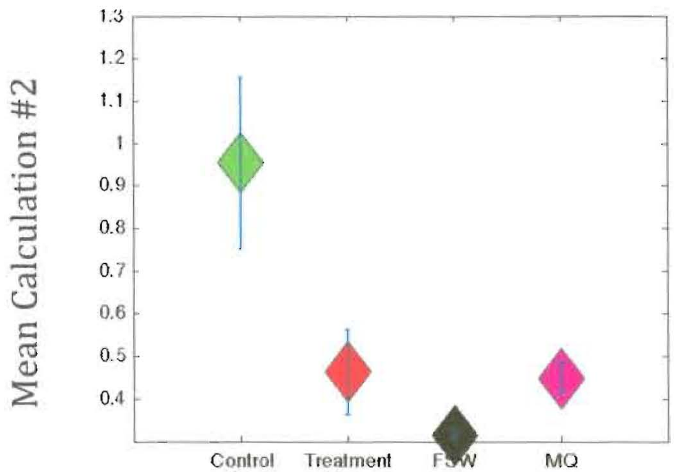

Figure 31. Mean calculation #1 values across sample types.

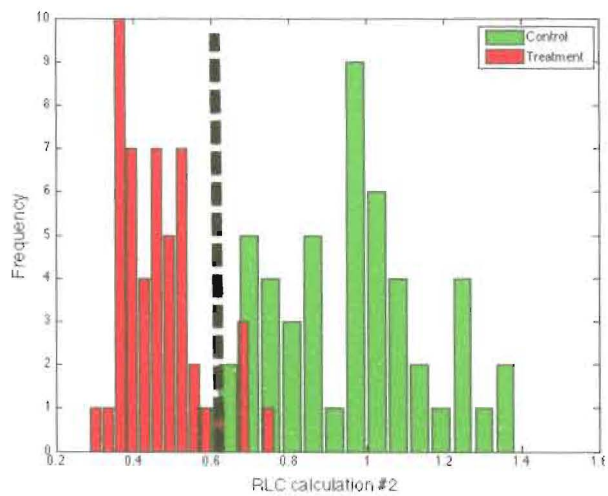

Figure 32. Histogram of calculation #1 values for control and treatment samples.

Concentration Experiments

Concentration experiments were conducted to compare whether or not the dark-adapted F_v/F_m signal could be improved by modifying the samples before algorithm analysis. The results of the concentration experiments (Figure 33) suggest that concentrating the control sample has no effect on increased dark-adapted F_v/F_m signal (*pairwise t-tests for each concentration method*, p-value > 0.05). Furthermore, using the criteria established with the series of algorithms, such as an F_v/F_m ratio above 0.5, a slope above 0.02, and a calculation #2 higher than 0.65, all of the concentrated control samples fell well above those thresholds (Table 6), reinforcing biological activity presence. Results from concentrating treatment samples showed the majority of the samples fell below all of the established biological activity thresholds. However, sample differed from the rest (Figure 33). When applying the threshold criteria framework to that particular sample, it falls below 2 of the 4 criteria, so viability was determined to be unlikely. Even thought the rest of the concentration results fell below the thresholds in Table 6, having multiple ways to assess the data such as the criteria thresholds, likely gives a better assessment of ambiguous samples.

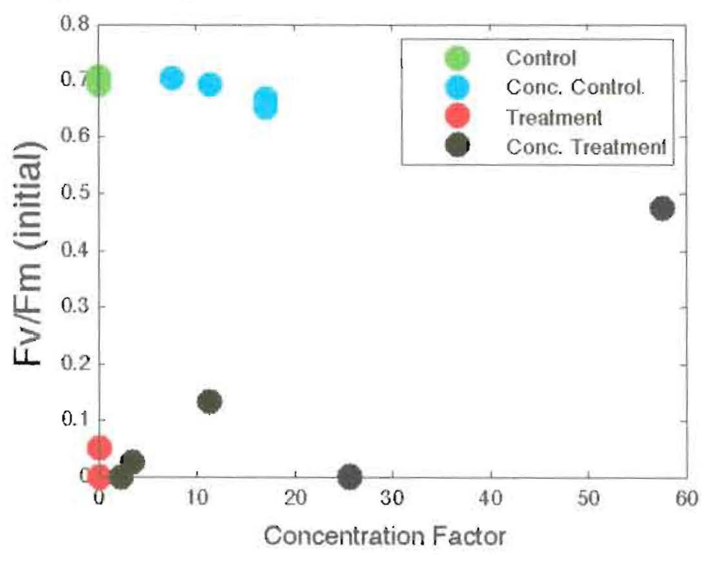

Figure 33. Initial F_v/F_m response to concentration methods.

Table 6. Results from the concentration experiments.

Sample	Replicate	Method	Conc. Factor	F_v/F_m	slope	Calc #2	viable
Treatment	A		0	0	-0.002	0.35	No
	B		0	0.05	-0.001	0.44	No
	A	Centrif.	3.37	0.025	0.002	0.28	No
	B	Centrif	2.25	0.05	-1.46	0.41	No
	A	Filtration	11.3	0.132	0.002	0.3	No
	B	Filtration	25.6	0	-0.015	0.27	No
	A	Centrif	129.7	0.475	-0.015	0.8	Unlikely
	B	Centrif	25.6	0	-0.001	0.18	No
Control	A		0	0.707	0.081	1.05	Yes
	B		0	0.693	0.063	1.06	Yes
	A	Centrif.	7.59	0.703	0.069	1.01	Yes
	B	Centrif.	11.3	0.693	0.05	1.02	Yes
	A	Filtration	17.08	0.668	0.06	0.98	Yes
	B	Filtration	17.08	0.652	0.04	0.95	Yes

Series of Algorithms Method Assessment

In order to assess how the series of algorithms and established thresholds can be useful in situations where cell concentrations are low, the treatment samples were used as a test case to see whether the four criteria together were better at determining lack of biological response than the typically used F_v/F_m ratio alone. A visual summary of the four criteria are illustrated in Figure 34 below: 1) dark-adapted F_v/F_m ratio, 2) quality assessment from peak 1, 3) F_o' baseline slope and the 4) the 'brute-force' signal analysis integrated over the full 13 min RLC.

1. Dark adapted F_v/F_m ratio (<0.2).
2. Peak 1 quality assessment.
3. Baseline Slope (slope < 0.02).
4. Calculation #2 'bulk' characterization (value < 0.65).

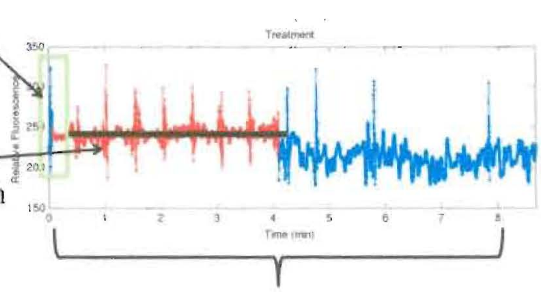

Figure 34. Illustrated summary of the four criteria established for indicating viability.

The summary of the results indicate that the final assessment using all the criteria is that the majority of the treatment samples showed no biological response, according to each criterion. Of the 50 samples, 7 had one or two out of the four criteria fall above the threshold. In those cases, the series of algorithms point out the samples that are at that edge of lack or presence of biological activity and indicate that those particular samples should be scrutinized further (looking at which thresholds were exceeded) and possibly be rerun to look at the replicate results for determination. By having this data analysis process, it can help give more information as to which samples are at that edge and what type of follow up is needed to make a judgment regarding presence or lack of biological activity.

Table 7. Treatment viability assessment using established criterion and thresholds.

	Biological Response Threshold	
Criteria	**Below**	**Above**
1. $F_v/F_m < 0.2$	48	2
2. Peak #1 signal/noise <50	47	3
3. Slope < 0.02	50	0
4. Calculation #2 < 0.65	46	4

Discussion

Custom Algorithms vs WALZ

In order to determine whether WALZ's undisclosed method of calculating the fluorescence parameters could be improved, the same parameters were calculated using two different methods after processing the rapid light curve through a series of calculations algorithms. The results from testing the differences in initial, dark-adapted F_v/F_m ratios from the programmed algorithms to WALZ standard data showed that the values were not significantly different from one another within sample types. Both values, F_o or F_m, are used in the calculation of F_v/F_m, so if programmed algorithms values varied significantly from WALZ, the ratios would also differ significantly. Since the initial F_v/F_m ratios were essentially the same, it is concluded that both the F_o and F_m determining algorithm are on par with what WALZ selects as those values. This also means that subsequent measurements that would utilize the F_v/F_m ratios (such as calculating the electron (ETR) transport rate during the RLC) would also be comparable to the WALZ calculations as well.

Quantifying 'No Biological Response'

A series of custom algorithms to analyze rapid light curve trends resulted in four criteria that aid in the differentiation between healthy and stressed, ballast water treatment samples, the latter characterized by low chlorophyll concentrations: specifically, 1) dark-adapted F_v/F_m ratio, 2) corresponding saturation peak signal to noise ratio, 3) F_o baseline slope value and 4) 'bulk' characterization using the brute-force calculation #2 method.

Plant and algal physiologists alike typically prefer the dark-adapted F_v/F_m ratio, alone, as the main indicator for biological activity since it is a quick measurement (< 2 seconds); however there is potential for ambiguity in those ratios when measuring close to the detection limit of the instrument (i.e. false peak due to noise resulting in an F_v/F_m greater than zero). Peak detection is therefore essential for determining whether the signal (F_v/F_m ratio) is an accurate representation of presence or absence of biological activity. Two peak criteria methods were developed in order to corroborate the signal; the dark-adapted peak F_v/F_m ratio (after algorithm processing) and dark-adapted peak signal to noise ratio. For the first criterion, the range of different the F_v/F_m ratios for each sample type were plotted to determine the threshold between presence and absence of biological activity. The results suggest that treatment samples with low chlorophyll and no biological response (including blank water samples) had F_v/F_m ratios of 0.2 or lower. It is important to note that while it is expected for blank samples like FSW and Milli-Q to have F_v/F_m ratios of zero, some of the replicates had ratios around 0.1. This phenomenon typically occurs when samples are measured on a higher sensitivity level and additional noise is introduced as a result. This reinforces the need to quantify the 'noise' in order to gather confidence in the presumed level of detection. Setting a threshold at an F_v/F_m of 0.2 is one criterion established for the purpose of quantifying the absence of biological response. The next step for assessing the peak was to qualitatively assess the 'quality' of the peak signal; i.e. determine whether it is distinguishable from the baseline noise.

The second criterion method involved taking the signal to noise ratio (SNR) of the dark-adapted peak to corroborate the calculated F_v/F_m ratio; i.e. if the SNR was low and the F_v/F_m ratio was high, it would indicate ambiguity in the sample measurement. The reason this second criterion, in contrast to the other three, resulted in a qualitative approach was due the to the overlap in the standard deviation of the means for the paired control and treatment samples, and the organism-free Milli-Q water. This overlap is likely due to the necessary increase of sensitivity (higher PM gains) in order to read these natural and treated seawater samples and creating additional variability within the 'noise' or baseline fluorescence. To be on the conservative side with regards to the noise issue, when empirically determining the absence of biological activity threshold of the dark-adapted peak, the threshold SNR value was established at 50 (where 94% of treatment samples fell below that value). By having two methods to corroborate the critical first measurement, the dark-adapted F_v/F_m ratio, it promotes confidence in establishing no biological response.

The third criterion was based on a general trend apparent for healthy samples during a rapid light curve (RLC); the baseline fluorescence is expected to increase as the actinic light increases. Therefore, the slope of that baseline increase was investigated as a way to quantify when samples no longer showed a biological response to the actinic light. The results also showed that live samples had measurable responses within the slope of the baseline fluorescence itself. The live samples had higher positive slopes (~0.05), while the treated or regular water samples had slopes closer to zero. Intuitively this makes sense since the 'height' of the peak F_v, decreases during the actinic portion of the rapid light curve due to the higher proportion of closed PSII reaction centers from photochemical quenching (i.e. the minimum fluorescence increases and the maximum fluorescence decreases). If the sample does not contain viable cells with functional PSII apparatuses, then there will not be much, if any, fluorescent response to the increasing actinic light levels (like photochemical quenching) meaning it will resemble a flat slope line. Control samples ranged from 0.03 – 0.05, but the rest of the water samples fell well below 0.02. This suggests that a good, conservative threshold criterion for quantifying no biological response using the baseline slope criterion is for the slope to be below 0.02.

The fourth criterion involved assessing each RLC in its entirety by proposing a method that would grossly characterize the presence or absence of a biological response; a 'brute-force' statistical parameter. Two methods were proposed and tested to determine which would yield a better numerical indicator for distinguishing between control and treatment samples. The results from assessing the RLC using calculation #1 and #2 revealed that using calculation #2 would be the most reliable criterion of the two. Calculation #1 revealed that while the individual control mean values were significantly different from the treatments, the standard deviations overlapped the treatment and blank water samples. Furthermore, when the frequency of observed values was plotted on a histogram, there was a large percentage of overlap between the controls (40%) and the treatment samples below the calculation #1 value of 50. This may be due to the method that this calculation is based on; taking an overall 'signal to noise ratio' of the rapid light curve. This means it is subject to the increased baseline variability (i.e. noise) when rapid light curves are measured at higher sensitivity levels. On the other hand, Calculation #2 results, strongly demonstrate that this method would be useful for quantifying no biological response; control and treatment samples exhibit a separation around a value of 0.65. Overall, using calculation #2 provided the best 'brute-force' method to numerically distinguishes samples with and without a biological response.

By analyzing different aspects of the rapid light curve trends and assessing whether treatment samples are at the edge of showing a biological response, the series of algorithms approach revealed samples that may need to be investigated further; a couple of treatment samples exceeded the minimum no biological response threshold in more than one category. While the thresholds were established with a conservative approach, the proposed algorithms do show the robust nature of establishing more confidence in the samples that fell below those thresholds 'no biological'; i.e. having more than one criterion satisfied leads to more certainty that the sample does not contain biologically active cells (or contains them at such a low concentration that the water would pass ballast water compliance testing purposes).

The results of the PAM RLC experiments have ground truth established from other complementary experiments conducted by the Biological Oceanography lab (using the same samples as the PAM RLC experiments for this thesis) using Environmental Technology Verification Program (ETV) protocol. Figure 35 shows one example of side-by-side analysis of the MPN and cell-specific FDA 'live' tagging procedure to determine numeric live concentrations of the 10-50 μm phytoplankton size class after UV-based concentration counts. However, UV-treated water revealed significant 'false positive' live cell concentrations. Furthermore, the FDA-based method failed to pass ballast water discharge in the majority of the trials, while MPN experiments showed a failure to detect growth in incubation tubes for a little more than half the UV-treatment samples (which resulted in approaching the method detection limit). The final results of the MPN experiments 'passed' the ballast water discharge standards (Welschmeyer and Maurer 2014). This means the samples showed no biological response and corresponds to the results of the treatment samples that were analyzed for biological activity using the established threshold criteria. However, it is important to note that many aquatic photoautrophic and even heterotrophic microorganisms from natural samples are difficult to grow in a laboratory environment, making estimates of viable biomass conservative at best. This highlights the need for an additional method, such as the proposed series of algorithms method to offer an additional method for assessing viability; the MPN-based method is conservative while the FDA-method suggests overestimation. While the PAM RLC measurements are not quantitative in the numeric counts that are typical for regulations, some of the treatments did have a couple biological thresholds exceeded. Those particular samples would need to be investigated further, especially since they have not exhibited growth during the MPN experiment process.

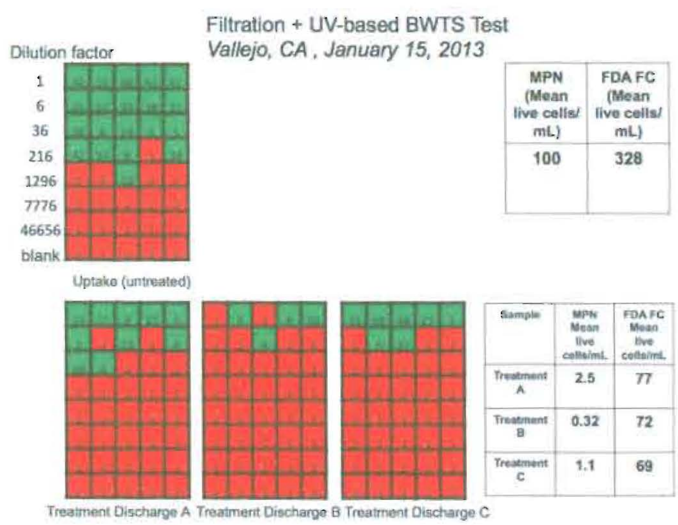

Figure 35. Results from a report by Welschmeyer and Maurer 2014 showing the results of the side-by-side analysis of the MPN- and FDA-based methods for quantifying live cells.

Concentration Experiments

A few concentration experiments were conducted in order to determine whether concentrating treatment samples (low F_v/F_m values) would increase the signal by increasing the chance that additional live cells existed and would be detected by the PAM. Two methods of concentration were used, centrifugation and gravity filtration. The majority of the results indicate that concentrating samples by centrifugation or gravity fed filtration do not increase the F_v/F_m ratio. However one concentrated treatment sample (concentrated ~60 fold) had an initial F_v/F_m of 0.45, suggesting the possibility for viability. When the series of algorithms criteria thresholds for no biological activity were applied to that sample though, it has a baseline slope of 0. While there appeared to be peaks in the data (leading to a higher calculation #2 value around 0.8), the baseline fluorescence slope suggests the sample did not respond to the actinic light, which is an expected response for a viable sample. Furthermore the other replicate, concentrated via gravity filtration at a lower concentration factor, did not surpass any of the criteria, so it is unlikely that the sample is viable given all aforementioned reasons. This outlier though does confirm that having multiple criteria for assessing RLC data, especially samples that seem ambiguous or contradictory, is crucial for making an assertion about the viability of the sample.

Conclusions

A series of algorithms were developed to process rapid light curves common to PAM fluorometry procedures. This resulted in the establishment of four criteria and corresponding thresholds for quantifying the absence of a biological response, for those cases in ballast water treatment testing where biological activity was intended to be reduced. Furthermore, the results from the proposed algorithms illustrate that complementary criteria thresholds are likely better at corroborating lack of cell biological response by the dark-adapted F_v/F_m ratio alone. The described criteria may be a positive step toward using the PAM fluorometer for the quantification of no biological response for the phytoplankton size class when used for ballast compliance testing. Furthermore, the program itself is flexible for the algal and plant physiologists alike that are interested in quantifying realistic assessments of photoautotrophic physiological status since the WALZ software output is limited to the fluorescence and quenching parameters.

The benefits of the series of algorithms (Table 8) include 1) calculation of peak parameters, 2) RLC plots, 3) individual peak plots, 4) characterization of sample biological activity, and 5) automated file processing with user friendly output. While both the proposed series of algorithms and WALZ can calculate peak parameters, the algorithms have the advantage of disclosed methods behind those calculations. Again both options are able to plot the rapid light curves, however it is contingent upon starting the chart in WALZ to even get this information in the first place. Once the chart is completed, the series of proposed algorithms has the added benefit of being able to normalize and smooth the fluorescence data for direct comparison to other samples. Third, the series of algorithms plots individual peaks in a color-coded manner. Fourth, one of the biggest draws to using these series of algorithms is that it characterizes the sample for biological activity using the 4 criteria and thresholds established from this thesis. And finally, the other crucial aspect of the proposed series of algorithms includes the automated ability to input WALZ files, analyze the data in the aforementioned ways, store and present the data in a user-friendly form.

Table 8. Summary of benefits using the series of algorithms compared to WALZ.

Capabilities	Series of Algorithms	WALZ
Calculate peak parameters (F_o, F_m, F_v/F_m)	Yes and disclosed methods	Yes
Plot entire RLC*	Yes and normalized with color code plots.	Yes, *but only if start chart
Plot individual peaks	Yes and color coded peaks.	No
Characterizes sample for biological activity	Yes and four criteria established.	No
Automated fluorescence data processing	Yes	No

References

Carlton, J and Geller, J. 1993. Ecological Roulette: The global transport of nonindigineous Marine Organisms. *Science* 261(5117): 78-82.

Dobroski, N., Scianni, C., Gehringer, D., and Falkner, M. 2009. Assessment of the efficacy, availability and environmental impacts of ballast water treatment systems for use in California waters. Prepared for the California State Legislature, California State Lands Commission, 173p.

Falkowski, P, Wyman, K., Ley, A., and Mauzerall, D. 1986. Relationship of steady-state photosynthesis to fluorescence in eucaryotic algae. *Biochimica Biophysica Acta* 849: 183-192.

First, M. and Drake, L. 2012. Performance of the human "counting machine": evaluation of manual microscopy for enumerating plankton. *Journal of Plankton Research* 34(12): 1028-1041.

Hallegraeff, G. and Bolch, C. 1991. Transport of toxic dinoflagellate cysts via ships ballast water. *Marine Pollution Bulletin* 20(27).

Hallegraeff, G. and Bolch C. 1992. Transport of diatom and dinoflagellate resting spores in ships ballast water-implications for plankton biogeography and aquaculture. *Journal of Plankton Research* 14(8).

Hamilton, R.D., and Holm-Hansen, O. 1967. Adenosine triphosphate content of marine bacteria. *Limnology and Oceanography* 12: 319-324.

Hendrickson, L., Furbank, R., and Chow, W. 2004. A simple alternative approach to assessing the fate of absorbed light energy using chlorophyll fluorescence. *Photosynthesis Research* 82: 73-81.

International Maritime Organization. 2004. Convention BWM/CONF/36 'International Convention for the Control and Management of Ships' Ballast Water and Sediments, 2004'.

Jordanov, V., Hall, D., and Kastner, M. 2003. Digital peak detector with noise threshold. *IEEE* 1: 140-142.

Kautsky, H. and Hirsch, A. 1931. Neue Versuche zur Kohlensäureassimilation. *Naturwissenschaften* 19, 964.

Karl, D. M. and O. Holm-Hansen. 1978. Methodology and measurement of adenylate energy charge ratios in environmental samples. *Marine Biology* 48: 185-197.

Karl, D. M. 1993. Adenosine triphosphate (ATP) and total adeninenucleotide (TAN) pool turnover rates as measures of energy flux and specific growth rate in natural populations of microorganisms, p. 483–494. In P. F. Kemp, B. F. Sherr, E. B. Sherr, and J. J. Cole [eds.], Handbook of methods in aquatic microbial ecology. Lewis Publishers.

Kolber, Z., Prásil, O., and Falkowski, P. 1998. Measurements of variable chlorophyll fluorescence using fast repetition rate techniques: defining methodology and experimental protocols. *Biochimica Biophysica Acta* 1367: 88–106.

Kropuenske, L., Mills, M., van Dijken, G., Bailey, S., Robinson, D., Welschmeyer, N., and Arrigo, K. 2009. Photophysiology in two major Southern Ocean phytoplankton taxa: Photoprotection in Phaeocystis Antarctica and Fragilariopsis cylindrus. *Limnology and Oceanography* 54(4): 1138-1151.

Kroon, B., Prézelin, B., and Schofield, O. 1993. Chromatic regulation of quantum yields for photosystem II charge separa- tion, oxygen evolution, and carbon fixation in Heterocapsa pygemaea (Pyrrophyta). *Journal of Phycology* 29: 453–462.

Ma, Z., Gao, K., Li, W., Xu, Z., Lin, H., and Zheng, Y. 2011. Impacts of chlorination and heat shocks on growth, pigments and photosynthesis of *Phaeodactylum tricornutum* (Bacillariophyceae). *Journal of Experimental Marine Biology and Ecology* 397: 214-219.

Mackey, K., Paytan, A., Frossman, A. and Bailey, S. 2008. A photosynthetic strategy for coping in a high-light, low nutrient environment. *Limnology and Oceanoraphy* 53(3).

Maxwell, K. and Johnson, G. 2000. Chlorophyll fluorescence-a practical guide. *Journal of Experimental Botany* 51(345).

Reavie, E., Cangelosi, A., and Allinger, L. 2010. Assessing ballast water treatments: Evaluation of viability methods for ambient freshwater microplankton assemblages. *Journal of Great Lakes Research* 36(3).

Schreiber, U., Bilger, W., Hormann, H., and Neubauer, C. 1998. *Photosynthesis: A Comprehensive Treatise*. Ed. A.S. Raghavendra. Cambridge: Cambridge University Press. Print.

Schreiber U., Hormann H., Neubauer C., and Klughammer C. 1995. Assessment of photosystem II photochemical quantum yield by chlorophyll fluorescence quenching analysis. *Australian Journal of Plant Physiology* 22: 209–220.

Serôdio J, da Silva J., and Catarino F. 1997. Nondestructive tracing of migratory rhythms of intertidal benthic microalgae using in vivo chlorophyll fluorescence. *Journal of Phycology* 33: 542 – 553.

Shapiro, H. 2003. *Practical Flow Cytometry*. New Jersey: John Wiley & Sons, Inc. Print.

Smith, S. 1997. *Digital Signal Processing: A Practical Guide for Engineers and Scientists.* San Diego: California Technical Publishing. Print.

Steinberg, M., First, M., Lemieux, E., Drake, L., Nelson, B., Kulis, D., Anderson, D., Welschmeyer, N., and Herring, P. 2012. Comparison techniques used to count single-celled viable phytoplankton. *Journal of Applied Phycology* 24(4): 751-758.

Suggett, D., Moore, M., Oxborough, K., and Geider, R. 2005. Fast Repetition Rate (FRR) Chlorophyll a Fluorescence Induction Measurements. Chelsea Technologies Group, Surrey p.1-7.

Taneja, H. 2008. *Advanced Engineering Mathematics: Volume 2.* New Delhi, India: I. K. International Pvt Ltd. Print.

Townsend, David. 2012. *Oceanography and Marine Biology: An Introduction to Marine Science*. Sunderland: Sinauer Associates, Inc. Print.

U.S. Coast Guard. 2009. Standards for living organisms in ships' Ballast water discharged in U.S. waters (notice of proposed rulemaking). *Federal Register*, 74, 44632-44672.

U.S. Environmental Protection Agency. 2011a. Draft National Pollutant Discharge Elimniation System (NPDES) General Permits for Discharges Incidental to the Normal Operation of a Vessel, EPA-HQ-OW-2011-01141 and EPA-HQ-2011-0150, United States Environmental Protection Agency, Washington, DC.

U.S. Environmental Protection Agency. 2011b. Efficacy of Ballast Water Treatment Systems: a Report by the EPA Science Advisory Board, EPA-SAB-11-009, United States Environmental Protection Agency, Washington, DC.

Van Kooten, O. and Snel, J. 1990. The use of chlorophyll fluorescence in nomenclature in plant stress physiology. *Photosynthesis Research* 25: 147-150.

Vernotte, C., Etienne, A., and Briantais, J-M. 1979. Qunching of the system II chlorophyll fluorescence by the plastoquinone pool. *Biocimica Biophyicia Acta* 545: 519:527.

Welschmeyer, N. and Maurer, B. 2014. Rationale for the use of most probable number (MPN) technique in the evaluation of UV-based Ballast Water Management Systems, Moss Landing Marine Laboratories, Moss Landing, CA.

Welschmeyer, N. 1994. Fluorometric analysis of chlorophyll a in the presence of chlorophyll b and pheopigments. *Limnology and Oceanography* 39: 1985-1992.

White, A. and Critchley, C. 1999. Rapid light curves: A new fluorescence method to assess the state of the photosynthetic apparatus. *Photosynthesis Research* 59: 63-72

sysvars_.m

```
MAINdir = ['/Users/youngautotech/Documents/MATLAB/thesis_data/'];
wildcard = [ ];
HTMLpamdir = [MAINdir 'processed_hach_data/HTML_data/'];
PRpamdir = [MAINdir 'processed_hach_data/imported_pcfs_reports/'];
FPpamdir = [MAINdir 'processed_hach_data/combined_prs/'];
```

getfname.m

```
function File = getfname(Path)
% GETFNAME  - Get the name of all specified files in a directory
% Use as: Fname = getfname(Path)
% Inputs: Path = Includes directory to look in and filename filter
%                (Default = '/*.m')
% Output: Fname  = Character array of filenames
% B. Schlining
% 10 Jul 97
if nargin < 1
   Path = '*.m';
end
FDat = dir(Path);
[r c] = size(FDat);
if r == 0
   File = [];
else
   File = char(FDat.name);
end
```

rlcanalysis_.m

```
function rlcanalysis_(name,option);
%rlcanalysis_.m
%This function's purpose is to search for .pcf files within a specified
%directory. It searches through the directories within that location for the
%correct wildcard name and then within that directory to extract the .pcf
files.
%Use As: rlcanalysis_(name,option)
%Input:  name   = '12UT8' as example, leave off any extensions.
%        option = 1 Reads in the pcf and saves in .mat file
%        option = 2 Combines the csv and pcf mat files into one file
%        option = 3 Smooths the data, finds the peaks
%        option = 4 Plots Fv, FvFm, & ETR calculations for both this program
%                   and WALZ WinControl's calculations.
%        option 5 = Calculates stat information like stdev, var and slope
%                   of baseline and peak data.
%        option 6 = Allows for comparison of uptake to treatment samples
%                   (compares calculations and peak to peak).
%        option 7 = Creates html output for individual charts or comparison
%                   of uptake to treatment charts.
%        option 8 = Creates an index of html chart files that have been
%                   created.
%Output: option 1 = name1.mat file for pcf, name2.mat for csv created in
%                   imported_pcfs_reports directory (see sysvars_.m)
%        option 2 = name_pr.mat file created in combined_prs directory.
%        option 3 = Saves Files As:
```

```
%                    Smooths only: name_s.mat + figures in HTML_data
%                    Norm+smooths: name_ns.mat+ figures in HTML_data
%                    Peak,smooth : name_sfp.mat+ figures in HTML_data
%                    Peak,norm+sm: name_fp.mat+ figures in HTML_data
%         option 4 = Saves figures (.png) in HTML_data only
%         option 5 = Saves stat information in _st.mat file & baseline
%                    figures in HTML_data direcotry.
%         option 6 = Saves comparison information in file _cns or _cs.mat
%         option 7 = Saves html output page in HTML_data directory.
%         option 8 = Saves the index of html chart files.
%Notes:
%  (1) If you do not wish to save the data and just wish to look at it,
%      there is the option of going to the individual program and running it
%     by itself. This program is meant to be run assuming you just want to
      run the analysis or comparison step by step to examine the final html
      output.
%  (2) To learn more about a program, go to the comment section of that
%      .m file program, type 'help programname' in command window or review
%      'how to' manual (word doc).
%
%Created: 1 APR 2013 by N.Bobco
if option == 1;
    sysvars_;
    if ~exist(PRpamdir);
        mkdir(PRpamdir)
    end
    pcfname = [name '.pcf'];
    Pname = getfname(pcfname)
    read_pcf_(Pname,1);
    repname = [name '.csv'];
    Rname = getfname(repname)
    readcsv_(Rname,1);
elseif option == 2
    combine_pcfcsv_(name,1);
elseif option == 3
    disp('Do you want to normalize data to PM gain 15?')
    normal = input('Enter 1 for yes, 0 for no: ');
    smooth_noise_fnc(name,normal,1);
    find_peaks_fnc_(name,normal,1);
elseif option ==4
    disp('Do you want to analyze normalized+smoothed or smoothed only data?')
    normal = input('Enter 1 for normalized+smoothed, 0 for smoothed only: ');
    pamcompare_(name,normal,1);
elseif option ==5
    disp('Do you want to analyze normalized+smoothed or smoothed only data?')
    normal = input('Enter 1 for normalized+smoothed, 0 for smoothed only: ');
    stats_(name,normal,1);
elseif option ==6
    uptake = input('Enter name of uptake file with single quotes around name:
    ');
    treatment = input('Enter name of treatment file with single quotes around
    name: ');
    number = input('Enter the total number of charts you wish you compare:
    ');
    disp('Do you wish to analyze normalized+smoothed or smoothed only data?')
    normal = input('Enter 1 for normalized+smoothed, 0 for smoothed only: ');
    cyclecompare_(uptake,treatment,number,normal,1);
```

```
elseif option == 7
    name = input('Enter name of file you wish you output results to an html
    page: ');
    disp('Do you wish to output results for normalized+smoothed or smoothed
    only data?')
    normal = input('Enter 1 for normalized+smoothed, 0 for smoothed only: ');
      type = input('Enter 1 for individual chart output, enter 2 for
comparison chart output: ');
    mkhtmlpg_(name,type,normal,1);
elseif option == 8
    disp('Work in progress')
    mkopttables_
else
    disp('not a valid code')
end
end
```

read_pcf_.m

```
function read_pcf_(filename,option) %save all as .mat file to process, option
do plot to check display
%read_pcf_.m
%This function's purpose is to read in the .pcf (Pam Chart File) files and
%save the fluorescence and report data into matrix variables.
%Use As: read_pcf_('filename.pcf',1)
%Input:  filename = PAM light curve data file .pcf
%        option   = 1  save .matfile of read in pcf data with variables.
%                 = 0  not save data.
%Output: name1.mat with 'pamp' 'fluorescence' 'comment' and 'pamp_header'
%        saved in PRpamdir.
% N. Bobco 6 SEP 2013
if nargin == 0
    help(mfilename) %returns name of file running
    return
end
[FID, MESSAGE] = fopen(filename,'r','ieee-be');
if FID == -1
    disp('File could not be opened')
    return
end
numberOfRecords = fread(FID,1,'uchar') * 256 + fread(FID,1,'uchar');
pp = 1;
for record = 1:numberOfRecords
    comment(record,1:2048) = blanks(2048);
    n = 1;
    ch = 10;
    while ch ~= 0
        ch = fread(FID,1,'uchar');
        comment(record,n) = ch;
        n = n+1;
    end
    numberOfSatPulses = fread(FID,1,'uchar');
    for pulse = 1:numberOfSatPulses;
        f = fread(FID,1,'uchar') * 256 + fread(FID,1,'uchar');
        fm = fread(FID,1,'uchar') * 256 + fread(FID,1,'uchar');
        y = (fread(FID,1,'uchar') * 256 + fread(FID,1,'uchar'))/1000;
```

```
        etr = fread(FID,1,'uchar') * 256 + fread(FID,1,'uchar');
        qp = fread(FID,1,'uchar') * 256 + fread(FID,1,'uchar');
        qn = fread(FID,1,'uchar') * 256 + fread(FID,1,'uchar');
        npq = fread(FID,1,'uchar') * 256 + fread(FID,1,'uchar');
        offset = fread(FID,1,'uchar') * 256 + fread(FID,1,'uchar');
        par = fread(FID,1,'uchar') * 256 + fread(FID,1,'uchar');
        pamp(pp,:) = [record f fm y etr qp qn npq offset par]; %pamp = pam
parameters
        pp= pp + 1;
        %fprintf('Offset: %d, F: %d, Fm: %d, Y: %d, PAR: %d, ETR: %d, qP: %d,
qN: %d, NPQ: %d\n', offset, f, fm, y, par, etr, qp, qn, npq);
        pamp_header = char('record', 'F', 'Fm', 'Yield', 'ETR', 'qP', 'qN',
'NPQ', 'Offset','PAR');
    end
    chartResolution = fread(FID,1,'uchar');
    chartLength = fread(FID,1,'uchar') * 256 + fread(FID,1,'uchar');
    %fprintf('Ft: ');
    fluorescence(1:32768,record) = NaN;
    for chartData = 1:chartLength
        ft = (fread(FID,1,'uchar') * 256 + fread(FID,1,'uchar'))-4096;
         if ft > 32767
            ft = 32768;
        end
        if ft == -4096
            ft = NaN;
        end
        fluorescence(chartData,record) = ft;
    end
end
NAME = char(filename(1:end-4));
sysvars_;
if ~exist(PRpamdir)
    mkdir(PRpamdir)
end
if option == 1
    save([PRpamdir NAME
'_1.mat'],'comment','pamp_header','pamp','fluorescence')
end
fclose(FID) %put at end need to close file
```

readcsv_.m

```
function [RAW,data] = readcsv_(FILE,option);
% readcsv_.m
%This function is for importing the entire report csv file.
% Use As: [RAW,data] = readcsv_(FILE,option);
% Input:  FILE = 'filename.csv'
%         option = 1 : Choose to save the read in csv file
% Output: name2.mat file. Example of PAM csv report contents
%     No. ,Mark,    H:M:S,    D.M.Y,    F,  Fm',   Yield,   ETR,   PAR,
Temp,  Batt,   Fo', int.T,    qP,     qN,    NPQ
%    'PAM-CONTROL (Water) SETTINGS 12:12 20.NOV 12
'MF=3;SI=10;SW=0.8;AI=6;AW=0:10;AF=1.00;PG=24;MA=8;FOW=112;FI=6;FW=0:10;EF=0.
84;FO=271;AA=12;CT=0:20;CI=1;LW=0:30;LI=3;ID=0:40;IW=0:20;TO=0.0;TG=1.00;LO=0
;LG=1.00;
%    'LC 1 12:15:58 20.11.12
%     3225,  1 , 12:15:58, 20.11.12,  500,  500,   0.000,   0.0,     0,
```

```
0.0,  13.7,     0,    21, 0.000, 0.000, 0.000
%          RAW: shows original data with the automatic empty cells
%          raw: shows above example, with empty cells removed.
% 28 JUL 2013. N.Bobco
if nargin == 1
    option = 0;
end
sysvars_;
if ~exist(PRpamdir)
    mkdir(PRpamdir)
end
NAME = char(FILE(1:end-4));
fid = fopen(FILE,'rt');
STR = fread(fid,inf,'char');
fclose(fid);
STR = char(STR)';
indx = find(abs(STR) == 10);
STR(indx) = char(13);
STR = [char(13) STR char(13)];
retind = find(abs(STR) == 13);
counter = 0;
counters = 0;
PG = NaN;
LC = NaN;
%can add more variables here from settings just do the same as above and
%incorporate into first and final if else loop (see below).
for k = 1:length(retind)-1
    line = STR(retind(k)+1:retind(k+1)-1);
    if length(findstr(line,','))>10
        line = [',' line];
        indx = find(abs(line) == abs(','));
        for j = 1:length(indx)-1
            tmp = line(indx(j)+1:indx(j+1)-1);
            tmp = strrep(tmp,':',' ');
            num = str2num(tmp);
            if isempty(num)
                RAW{k,j} = tmp;
            else
                RAW{k,j} = num;
            end
        end
        RAW{k,25} = counter;
        RAW{k,26} = LC;
        RAW{k,27} = PG;
    elseif  length(findstr(line,';'))>15
        line = [';' line];
        indx = find(abs(line) == abs(';'));
        for j = 1:length(indx)-1
            tmp = line(indx(j)+1:indx(j+1)-1);
            tmp = [strrep(tmp,'''','') ';'];
            eval(tmp)
            RAW{k,j} = tmp;
        end
        counter = counter +1;
    elseif length(findstr(line,' '))<5
        line = [' ' line];
        indx = find(abs(line) == abs(' '));
```

```
        for j = 1:length(indx)-1
            tmp = line(indx(j)+1:indx(j+1)-1);
            tmp = strrep(tmp,':',' ');
            num = str2num(tmp);
            if isempty(num)
                RAW{k,j} = tmp;
            else
                RAW{k,j} = num;
                if j == 2
                    LC = num;
                end
            end
            counters = counters +1;
        end
        RAW{k,25} = counter;
        RAW{k,26} = LC;
        RAW{k,27} = PG;
    end
end
r = 1;
for R = 1:size(RAW,1) %rows
    if ~isempty(RAW{R,1})
        for C = 1:size(RAW,2) %columns
            raw{r,C} = RAW{R,C};
        end
        r = r+1;
    end
end
setting=counter/24;
record=counters/4;
clear RAW
RAW = raw;
clear raw
r = 1;
for R = 1:size(RAW,1) %rows
    for C = 1:size(RAW,2) %columns
        if isstr(RAW{R,C})
            tmp = deblank(RAW{R,C});
            tmp = fliplr(deblank(fliplr(tmp)));
            if length(abs(tmp)) == 1
                data(R,C) = abs(tmp);
            else
                tmp = RAW{R,C};
                tmp = strrep(tmp,'.',' ');
                num = str2num(tmp);
                if ~isempty(num)
                    num(3) = num(3)+2000;
                    data(R,C) = datenum(num(3),num(2),num(2));
                else
                    data(R,C) = NaN;
                end
            end
        elseif isempty(RAW{R,C})
            data(R,C) = NaN;
        elseif length(RAW{R,C}) > 1
            if C == 3
                tmp = RAW{R,C};
```

```
                data(R,C) = tmp(1)+tmp(2)./60+tmp(3)./60./60;
            else
                data(R,C) = NaN;
            end
        else
            data(R,C) = RAW{R,C};
        end
    end
    r = r+1;
end
indx = find(isnan(data(:,1)));
data(indx,:) = [];
data(:,16:24) = [];
if option == 1
    save([PRpamdir NAME '_2.mat'], 'data') %r for report
end
end
```

combine_pcfcsv_.m

```
function [mydata, report, settings, pamp, fluorescence] =
combine_pcfcsv_(name,option) %add chart and option once this works.
%combine_pcfcsv_.m
%This function is for importing the chart pcf and report csv files and
%combining them to make one combo matfile.
% Use As: [comment, pamp_header, fluorescence, pamp] =
combine_pcfcsv_(Fname,chart,option);
% Input:  nname = 'filename.mat'
%         option = 1 : Choose to save the combined pcf/csv mat file
%                  0 : Choose to NOT save the data
% Output: name_pr.mat in FPpamdir directory with 'settings', 'pamp and
%         'fluorescence' variables.
% 26 SEP 2013. N.Bobco
sysvars_;
if ~exist(FPpamdir)
    mkdir(FPpamdir)
end
if nargin == 1
    option = 0;
end
if option ==1
    matname = [name '_*.mat']
    Fname = getfname(matname)
    numfiles = 2;
    mydata = cell(1, numfiles);
    r = 1;
    for k = 1:numfiles
        for k = 1:numfiles
            mydata{r,k} = importdata(Fname(k,1:length(Fname)));
        end
    end
    pcf = struct2cell(mydata{1});
    report = mydata{1,2};
    settings = (mydata{2}(:,16:18));
    pamp = cell2mat(pcf(2));
    fluorescence = cell2mat(pcf(4));
```

```
    save([FPpamdir name '_pr.mat'],'settings','pamp','fluorescence') %pr for
combined pcf and report data
else disp('not a valid code')
end
end
```

smooth_noise_fnc.m

```
function [smoothnormdata] = smooth_noise_fnc(name,normal,option);
%smooth_noise_fnc.m
%This function's purpose is to either: smooth the noise only (peaks removed
%then replaced) or normalize the data to PM 15 and then smooth the noise
%(peaks removed then replaced in tact). It cycles through each column of
%fluorescence data for combined .pcf and .csv files. Requires file
%name_pr.mat to run this program. %Use As:
smooth_noise_fnc('filename',normal,option)
%
%Input:  name     = PAM Light Curve data file .mat extension (ex/ '12UT8'
%                   not '12UT8_pr.mat')
%        normal   = 1 For normalizing data to PM 15
%        normal   = 0 For raw, non normalized data.
%        option   = 1 Save .mat file and print figs to html_pam directory
%Prompted Inputs (in program):
%        chart    = 20 For 20 minute chart file
%Output: figure(1) Plot of raw or normalized data saved in HTML_data dir.
%        figure(2) Plot of smoothed noise saved in HTML_data dir.
%        FILE      name_s.mat for raw smoothed only
%                  name_ns.mat for normalized, smoothed' data.
%Created: 1 APR 2013 by N.Bobco
sysvars_;
if ~exist(HTMLpamdir)
    mkdir(pwd,HTMLpamdir)
end
if nargin == 1
    option = 0;
end
matname = [name '_pr.mat'];
Fname = getfname(matname)
load([FPpamdir Fname]);
if normal == 1
    normalize_(name,1);
    matname = [name '_ns.mat'];
    Fname = getfname(matname)
    load([FPpamdir Fname]);
    ext = '_ns';
    for c = 1:length(data(1,:))
        datacol = data(:,c);
        datasf = datacol;
        disp('Enter chart length in sequential order of chart number')
        chart = input('Enter chart length: ');
        if chart ==20  %20 min chart file
            figure(c)
            plot(fluorescence(:,c))
            xlabel('Time (min)', 'FontSize',14)
            ylabel('Relative Fluorescence', 'FontSize',14)
            title('Raw Data')
```

```
            set(gca,'XLim',[0 13000]) %set max value for 8 min chart length
            set(gca,'XTick',[0:1500:20000]) %about 1600 points per minute.
            set(gca,'XTickLabel',['0';'1';'2';'3';'4';'5';'6';'7';'8'])
            if option == 1
                print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir name ext '_r'
num2str(c) '.png'])
            end
            figure(c)
            plot(datacol)
            xlabel('Time (min)', 'FontSize',14)
            ylabel('Relative Fluorescence', 'FontSize',14)
            title('Normalized Data')
            set(gca,'XLim',[0 13000]) %set max value for 8 min chart length
            set(gca,'XTick',[0:1500:20000]) %about 1600 points per minute.
            set(gca,'XTickLabel',['0';'1';'2';'3';'4';'5';'6';'7';'8'])
            if option == 1
                print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir name ext
'_20n_' num2str(gcf) '.png']) %for smooth test
                set(gcf, 'PaperPosition',[0.25 2.5 4 3])
                print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir name ext
'_20n_' num2str(gcf) 'th' '.png'])
            end
            clear figure
            ERpeak = [1:80 750:800 1450:1546 2253:2405 3035:3136 3820:3909
4580:4650 5340:5385 6040:6195 6341:6409 7111:7204 8656:8740 11720:11761];
            datasf = [[1:length(datasf)]'  datasf]; %numbering datasf values
            datapeak = datasf; %where peak data occurs within the main data
index (defined in ERpeak)
            datasf(ERpeak,2) = NaN; %NaNs where peaks are,
span=50; %size of the averaging window may need to make bigger for 10 and 15
            window=ones(span,1)/span;
            datasf(:,2)=convn(datasf(:,2),window,'same'); %smoothing noise
            indx = find(isnan(datasf(:,2)));
            datasf(indx,2) = datapeak(indx,2);
            datacol = datasf(:,2);
            figure(c)
            plot(datacol, 'r')
            xlabel('Time (min)', 'FontSize',14)
            ylabel('Relative Fluorescence', 'FontSize',14)
            title('Normalized and Smoothed Noise')
            set(gca,'XLim',[0 13000]) %set max #points for 8 min chart length
            set(gca,'XTick',[0:1500:20000]) %about 1600 points per minute.
            set(gca,'XTickLabel',['0';'1';'2';'3';'4';'5';'6';'7';'8'])
            if option == 1
                set(gcf, 'PaperPosition',[0.25 2.5 8 6]);
                print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir name ext
'_20ns_' num2str(gcf) '.png']);
                set(gcf, 'PaperPosition',[0.25 2.5 4 3]);
                print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir name ext
'_20ns_' num2str(gcf) 'th' '.png']);
            end
            smoothnormdata(:,c) = datacol;
            chartindx(:,c) = chart;
            clear chart
            close all
        else
            disp('You did not enter a valid code. Chart length must be: 20')
```

```
        end
        if option == 1
            save([name '_ns.mat'],'smoothnormdata','chartindx',
'pamp','fval_w') %sn for normalized and smoothed
        end
    end
elseif normal == 0;
    for k = 1:length(pamp);
        Fo_w = pamp(k,2);
        Fm_w = pamp(k,3);
        FvFm_w = pamp(k,4);
        fval_w(k,:) = [k Fo_w Fm_w FvFm_w];
    end
    ext = '_s';
    for k = 1:length(data(1,:));
        data = fluorescence(:,k);
        datasf = data;
        disp('Enter chart length in sequential order of chart number')
        chart = input('Enter chart length: ')
        if chart ==20  %20 min chart file
            figure(c)
            plot(datacol)
            xlabel('Time (min)', 'FontSize',14)
            ylabel('Relative Fluorescence', 'FontSize',14)
            title('Raw Data')
            set(gca,'XLim',[0 13000]) %set max value for 8 min chart length
            set(gca,'XTick',[0:1500:20000]) %about 1600 points per minute.
            set(gca,'XTickLabel',['0';'1';'2';'3';'4';'5';'6';'7';'8'])
            if option == 1
                print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir name ext '_20r'
num2str(gcf) '.png']) %for smooth test
            end
            ERpeak = [1:80 750:800 1450:1546 2253:2405 3035:3136 3820:3909
4580:4650 5340:5385 6040:6195 6341:6409 7111:7204 8656:8740 11720:11761];
            datasf = [[1:length(datasf)]'  datasf]; %numbering datasf values
            datapeak = datasf; %where peak data occurs within the main data
index (defined in ERpeak)
            datasf(ERpeak,2) = NaN; %NaNs where peaks are, so can smooth the
data
            span=50;
            window=ones(span,1)/span;
            datasf(:,2)=convn(datasf(:,2),window,'same'); %smoothing noise
            indx = find(isnan(datasf(:,2)));
            datasf(indx,2) = datapeak(indx,2);
            datacol = datasf(:,2);
            figure(c+1)
            plot(datacol, 'r')
            xlabel('Time (min)', 'FontSize',14)
            ylabel('Relative Fluorescence', 'FontSize',14)
            title('Normalized and Smoothed Noise')
            set(gca,'XLim',[0 13000]) %set max value for 8 min chart length
            set(gca,'XTick',[0:1500:20000]) %about 1600 points per minute.
            set(gca,'XTickLabel',['0';'1';'2';'3';'4';'5';'6';'7';'8'])
            if option == 1
                print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir name ext
'_20rs' num2str(gcf) '.png']); %
            end
```

```
            smoothdata(:,c) = datacol;
            chartindx(:,c) = chart;
            clear chart
            close all
        else
            disp('You did not enter a valid code. Chart length must be: 20)
        end
        if option == 1
            save([name '_s.mat'],'smoothdata','chartindx', 'pamp','fval_w');
        end
    end
else
    disp('you did not enter a valid normal code (0 or 1 only)')
end
end
```

find_peaks_fnc.m

```
function [fval,lc] = find_peaks_fnc_(name,normal,option);
%find_peaks_fnc_.m
%This function finds peaks based on predetermined bin ranges for 20 minute
%chart length files only.  This function also calculates
%various PAM parameters that WALZ Wincontrol also contains: ex/ Fo, Fm,
%Fv & FvFm) for each peak. It also plots the entire data set as well as
%each individual 'peak' (or area where peak should be within bin).
%Requires mat file that has data either smoothed (name_s.mat) or normalized
%and smoothed (name_ns.mat). It will run through each chart contained in a
%file individually.
%Use As:  find_peaks_fnc_('filename',normal,option)
%Input:   name      =  Filename '12UT8' as example, leave off any extensions.
%         normal    =  1 Reads in normalized/smoothed data file.
%                   =  0 Reads in smoothed only data file.
%         option    =  1 Save figures in pngs and publish to html.
%Output:  figure (1)  Overview of light curve; control or treatment
%         figure (2)  Overview of peak. red is entire peak, blue points
%                     preceed the peak and green is after the end of the
peak.
%         figure (#)  Each individual peak from each chart file saved as
.pngs
%                     in HTML_data directory
%         FILES       Saves name_sfp.mat for smoothed only data.
%                     Saves name_fp.mat for norm+smoothed data.
%Created: 20 JUN 2013 by N.Bobco
close all
sysvars_;
if ~exist(HTMLpamdir)
    mkdir(pwd,HTMLpamdir)
end
if nargin == 1
    option = 0;
end
pp = 1;
if normal == 1
    load([FPpamdir name])
    namesave = name(1:end-4);
    ext = '_ns';
    M = length(smoothnormdata);
```

```
    N = length(smoothnormdata(1,:));
    peakindx = repmat(0,M,N);
    peakindx2 = repmat(0,M,N);
    peakindx3 = repmat(0,M,N);
    for c = 1:length(chartindx)
        chart = chartindx(c);
        data = smoothnormdata(:,c);
        if chart == 20
            hold off
            figure(c)
            multicolorplot(data,data,100:6150)
            set(gca,'XLim',[0 13000]) %set max value for 8 min chart length
            set(gca,'XTick',[0:1500:30000]) %about 1500 points per minute.
            set(gca,'XTickLabel',['0';'1';'2';'3';'4';'5';'6';'7';'8'])
            if option == 1
                print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir namesave ext
'_' num2str(chart) '_c' num2str(c) '.png'])
                set(gcf, 'PaperPosition',[0.25 2.5 4 3])
                set(gca, 'FontSize',8)
                print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir namesave ext
'_' num2str(chart) '_c' num2str(c) '_th.png']);
            end
            FH = figure;  %start of peak plots
            plot(data(:,1), '.b-')
            xlim([0 13000])
            hold on
            PBINS1 = [  1 700 1450 2200 3000 3775 4550 5300 6075 6300 7070
8600 11650];
            PBINS2 = [100 850 1650 2400 3200 3925 4700 5450 6200 6450 7220
8800 11850];
            for B = 1:length(PBINS1);
                er = PBINS1(B):PBINS2(B);
                peak = data(er,1);
                peakindx(er,c) = 1;
                peakindx2(er,c) = B;
                figure(2)
                [x,i] = nanmax(peak); %finds tallest peak, not all files have
first peak as tallest, ergo bins.
                eri = i-50:i;
                eri = eri(find(eri > 0));
                plot(er,peak,'r-')
                plot(er(eri),peak(eri),'.g-')
                title(['Peak ',num2str(B)],'FontSize',8);
                erl = eri;
                [x,il] = nanmax(abs(diff(peak(erl))));
                err = i+10:i+30;
                err = err(find(err > 0 & err <= length(er)));
            SN = NaN;
                   if ~isempty(err);
                    [x,ir] = nanmax(abs(diff(peak(err))));
                    figure(FH);
                    eri2 = erl(il)-1:err(ir)+10;
                    eri2 = eri2(find(eri2 > 0 & eri2 <=length(er)));
                    plot(er(eri2),peak(eri2),'.c-')
                    peakindx3(er(eri2),c) = 3;
```

```
                xlim([PBINS1(B)-20 PBINS2(B)+20])
                if ~isempty(eri2);
                    data(er(eri),2) = B;
                    Fo = peak(eri2(1));
                    Fo_a = mean(peak(eri(2):eri2(1))); %
                    Fm = data(er(i));
                    if length(eri2) >20
                        Fm_a = mean(peak(eri2(5:21)));
                        peakindx(er(eri2(5:21)),c) = 2;
                        hold on
                        plot(er(eri(2):eri2(1)),peak(eri(2):eri2(1)),'.g-
')

                        plot(er(eri2(5:21)), peak(eri2(5:21)), '.m-')
                    else
                        Fm_a=mean(peak(eri2(5:15)));
                        peakindx(er(eri2(5:15)),c) = 2;
                        hold on
                        plot(er(eri(2):eri2(1)),peak(eri(2):eri2(1)),'.g-
')

                        plot(er(eri2(5:15)), peak(eri2(5:15)), '.m-')
                    end
                    Fv = Fm-Fo;
                    Fv_a = (Fm_a-Fo_a);
                    FvFm = Fv./Fm;
                    FvFm_a = Fv_a./Fm_a;
                    SN = Fv_a./nanstd(peak(eri(2):eri2(1)))
                    PAR = pamp(pp,10);
                    ETR = PAR.*FvFm.*0.85*0.5;
                    ETR_a = PAR.*FvFm_a.*0.85*0.5;
                    data(eri,2) = B;
                    if option == 1
                        set(gcf, 'PaperPosition',[0.25 2.5 8 6])
                        set(gca, 'FontSize',10)
                        print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir
namesave ext '_' num2str(chart) '_c' num2str(c) '_p' num2str(B) '.png']);
                        set(gcf, 'PaperPosition',[0.25 2.5 2 1.5])
                        set(gca, 'FontSize',8)
                        print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir
namesave ext '_' num2str(chart) '_c' num2str(c) '_p' num2str(B) 'th.png']);
                    end
                end
            else
                Fo = NaN;
                Fo_a = NaN;
                Fm = NaN;
                Fm_a = NaN;
                Fv = NaN;
                Fv_a = NaN;
                FvFm = NaN;
                FvFm_a = NaN;
                ETR = NaN;
                ETR_a = NaN;
            end
            Fo_w = fval_w(pp,2); %WALZ Fo
            Fm_w = fval_w(pp,3); %WALZ Fm
            Fv_w = Fm_w-Fo_w;
            FvFm_w = pamp(pp,4); %WALZ Fv
```

```
                qP_w = (pamp(pp,6))./1000; %WALZ qP
                NPQ_w = (pamp(pp,5))./1000;%WALZ NPQ
                PAR = pamp(pp,10);
                ETR_w = pamp(pp,5)./10; %WALZ ETR
                fval(pp,:) = [c B Fo Fo_a Fo_w Fm Fm_a Fm_w Fv Fv_a Fv_w FvFm
FvFm_a FvFm_w SN]
                lc(pp,:) = [c B qP_w NPQ_w PAR ETR ETR_a ETR_w];
                Fo = NaN;
                Fo_a = NaN;
                Fm = NaN;
                Fm_a = NaN;
                Fv = Fm-Fo;
                Fv_a = NaN;
                FvFm = NaN;
                FvFm_a = NaN;
                SN = NaN;
                ETR = NaN;
                ETR_a = NaN;
                peak(1:eri(end)) = NaN;
                close all
                figure(FH)
                plot(data(:,1), '.b-')
                xlim([0 13000])
                hold on
                pp = pp+1;
            end
            pp = (13*c)+1;
        else
            disp('You did not enter a valid code. ')
        end
    end
    close all
    namesave = name(1:end-4);
    save([namesave '_fp.mat'], 'smoothnormdata', 'fval', 'chartindx', 'pamp',
'lc', 'fval_w', 'peakindx','peakindx2','peakindx3');
elseif normal ==0
    matname = [name '_s.mat']
    Fname = getfname(matname)
    load([FPpamdir Fname])
    ext = '_s';
    M = length(smoothdata);
    N = length(smoothdata(1,:));
    peakindx = repmat(0,M,N);
    peakindx2 = repmat(0,M,N);
    for c = 1:length(chartindx)
        chart = chartindx(c);
        data = smoothdata(:,c);
                if chart == 20
            figure(c)
            multicolorplot(data,data,100:6200)
            set(gca,'XLim',[0 13000]) %set max value for 8 min chart length
            set(gca,'XTick',[0:1500:30000]) %about 1500 points per minute.
            set(gca,'XTickLabel',['0';'1';'2';'3';'4';'5';'6';'7';'8'])
            if option == 1
                print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir name ext '_'
num2str(chart) '_c' num2str(c) '.png']) %name chart#, whole chart plotted
red/blue
```

```
            end
            FH = figure;  %start of peak plots
            plot(data(:,1), '.b-')
            xlim([0 13000])
            hold on
            PBINS1 = [  1 700 1450 2200 3000 3800 4550 5300 6100 6300 7070
8550 11650];
            PBINS2 = [100 850 1650 2400 3200 3950 4700 5450 6200 6450 7220
8800 11850];
            for B = 1:length(PBINS1);
                er = PBINS1(B):PBINS2(B);
                peak = data(er,1);
                peakindx(er,c) = 1;
                peakindx2(er,c) = B;
                figure(2)
                [x,i] = nanmax(peak); %finds tallest peak, not all files have
first peak as tallest, ergo bins.
                eri = i-50:i;
                eri = eri(find(eri > 0));
                plot(er,peak,'r-')
                plot(er(eri),peak(eri),'.g-')
                erl = eri;
                [x,il] = nanmax(abs(diff(peak(erl))));
                err = i+10:i+30;
                       err = err(find(err > 0 & err <= length(er)));
            if ~isempty(err);
                    [x,ir] = nanmax(abs(diff(peak(err))));
                    figure(FH);
                    eri2 = erl(il)-1:err(ir)+10;
                    eri2 = eri2(find(eri2 > 0 & eri2 <=length(er)));
                    plot(er(eri2),peak(eri2),'.c-')
                    xlim([PBINS1(B)-20 PBINS2(B)+20])
                    if ~isempty(eri2);
                        data(er(eri),2) = B;
                        Fo = peak(eri2(1));
                        Fo_a = mean(peak(eri(2):eri2(1)));
                        Fm = data(er(i));
                        Fm_a = mean(peak(eri2(5:21)));
                        Fv = Fm-Fo;
                        Fv_a = (Fm_a-Fo_a);
                        FvFm = Fv./Fm;
                        FvFm_a = Fv_a./Fm_a;
                        peakindx(er(eri2(5:21)),c) = 2;
                        PAR = pamp(pp,10);
                        ETR = PAR.*FvFm.*0.85*0.5;
                        ETR_a = PAR.*FvFm_a.*0.85*0.5;
                        hold on
                        plot(er(eri(2):eri2(1)),peak(eri(2):eri2(1)),'.g-')
                        plot(er(eri2(5:21)), peak(eri2(5:21)), '.m-')
                        data(eri,2) = B;
                        if option == 1
                            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir
name ext '_' num2str(chart) '_c' num2str(c) '_p' num2str(B) '.png']);
                            set(gcf, 'PaperPosition',[0.25 2.5 2 1.5])
                            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir
name ext '_' num2str(chart) '_c' num2str(c) '_p' num2str(B) 'th.png']);
                        end
```

```
                    end
                end
                Fo_w = pamp(pp,2);
                Fm_w = pamp(pp,3);
                Fv_w = Fm_w-Fo_w;
                FvFm_w = pamp(pp,4);
                qP_w = (pamp(pp,6))./1000;
                NPQ_w = (pamp(pp,5))./1000;
                PAR = pamp(pp,10);
                ETR_w = pamp(pp,5)./10;
                fval(pp,:) = [c B Fo Fo_a Fo_w Fm Fm_a Fm_w Fv Fv_a Fv_w FvFm
FvFm_a FvFm_w];
                lc(pp,:) = [c B qP_w NPQ_w PAR ETR ETR_a ETR_w];
                Fo = NaN;
                Fo_a = NaN;
                Fm = NaN;
                Fm_a = NaN;
                Fv = Fm-Fo;
                Fv_a = NaN;
                FvFm = NaN;
                FvFm_a = NaN;
                ETR = NaN;
                ETR_a = NaN;
                peak(1:eri(end)) = NaN;
                close all
                figure(FH)
                plot(data(:,1), '.b-')
                xlim([0 13000])
                hold on
                pp = pp+1;
            end
            pp = (13*c)+1;
        else
            disp('You did not enter a valid code. ')
        end
        close all
        save([name
'_sfp.mat'],'smoothdata','fval','Fname','chartindx','pamp','lc','fval_w',
'peakindx','peakindx2');
    end
end
```

multicolorplot_.m

```
function multicolorplot(x,y,range)
%multicolorplot.m
% Use As:   multicolorplot(x,y,#:#)
% Input:    x =  x axis indices ex: 1:length(data(:,1))
%           y =  y axis data data(:,1)
% Output:   figure (1)  Overview of light curve; control or treatment
%           figure (2)  Overview of peak. red is entire peak, blue points
%                       preceed the peak and green is after the end of the
peak.
%           figure (3)  Max positive peak in plot is locaion in reference to
%                       beginning of peak (index erl, same as eri
%           figure (4)  Max negative peak in plot is location in reference
%                       to end of peak (index err)
```

```
%           saved pngs and html export in directory html_pam
% Created: 21 AUG 2013 by N.Bobco
%Define indicies for light and dark parts
A=1:length(x);
%Dark parts of curve
B=y;
B(A(range(1)+1:range(end)-1))=NaN;
%Light parts of curve
C=y;
C(A(1:range(1)-1))=NaN;
C(A(range(end)+1:end)) =NaN;
plot(A,B,'b.-')
hold on
plot(A,C,'r.-')
xlabel('Time (min)', 'FontSize',14)
ylabel('Relative Fluorescence', 'FontSize',14)
title('Smoothed Light Curve Data','FontSize',18) %
end
```

stats_.m

```
function [bfline, stats, peakstats, percomp, perrecov] =
stats_(name,normal,option);
%stats_.m
%This function characterizes noise (STD, VAR) and gives idea of data quality,
%and calculates/plots slope of data baseline (best fit line). Uses data
%calculations from find_peaks_fnc_.m (requires name_fp.mat or name_sfp.mat
%files to exist) and original import data (name_pr.mat).
%Use As: stats_(name,normal,option)
%Input:  name   =   Filename'12UT8' as example, leave off any extensions.
%        normal = 1 Reads in normalized + smoothed name_fp.mat file
%               = 2 Reads in smoothed only name_sfp.mat file
%        option = 1 Saves all the Fv, FvFm, ETR plot comparisons in
%                   HTML_data directory as figures.
%Output: Figures    Saves baseline figures (.png) in HTML_data only
%        Files      Saves name_ns_st.mat for norm+smoothed data analysis
%                   Saves name_s_st.mat for smoothed only data analysis
%Created: 29 OCT 2013 by N.Bobco
if nargin == 1
    option = 0;
end
sysvars_;
if ~exist(FPpamdir)
    mkdir(FPpamdir)
end
if normal == 1
    matname = [name '_pr.mat']; %raw data
    Fname = getfname(matname)
    load([FPpamdir Fname]);
    matname = [name '_fp.mat']; %need index peak/noise info
    Fname = getfname(matname)
    load([FPpamdir Fname]);
    ext = '_ns';
    pp = 1;
    p = 1;
    PBINS1 = [1:13];
    for c = 1:length(fluorescence(1,:))
```

```
        indx = [fluorescence(:,c) peakindx(:,c)];
        noise = indx(find(indx(:,2) ==0)); %noise
        peaktop = indx(find(indx(:,2) ==1)); %top of peak
        chart = chartindx(:,c);
           if chart == 20
            noiseindx = [fluorescence(1:6150,c) peakindx(1:6150,c)];
            noise_vals = noiseindx(find(noiseindx(:,2) == 0));
            x = 1:length(noise_vals);
            x1 = x';
            poly = polyfit(x1,noise_vals,1);
            f = polyval(poly,x1); %evaluate equation and gives values
            bfline(c,:) = [c poly(1,1) poly(1,2)];
            figure(c)
            plot(x,noise_vals,'-r',x1,f,'ok')
            ylabel('Relative Fluorescence', 'FontSize',14)
            title('Light Curve Baseline Fluorescence (peaks, recovery
removed)')
            legend('Light Curve', 'best fit line')
            if option == 1
                set(gcf, 'PaperPosition',[0.25 2.5 8 6]);
                print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir name ext '_Fo_'
num2str(c) '.png']);
                set(gcf, 'PaperPosition',[0.25 2.5 4 3]);
                print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir name ext '_Fo_'
num2str(c) 'th.png']);
            end
            close all
        else
            disp('Check to see if chart values are correct')
        end
        sdr_n = nanstd(noise); %standard deviation
        allpeaks = [smoothnormdata(:,c) peakindx(:,c)];
        peaks_idx = allpeaks(find(peakindx(:,c) == 2));
        var_n = nanvar(noise); %variance
        var_pt = nanvar(peaktop); %not quite right....
        var_d = abs((var_n-var_pt)/var_n).*100;
        mean_n = nanmean(noise);
        %for overall data assessment
        mean_data = nanmean(fluorescence(:,c)); %take mean
        mean_peaks = nanmean(peaks_idx(:,1));
        SNR = mean_data./sdr_n; %signal to noise ratio
        SNR_t = mean_peaks./sdr_n;%with peak tops as average
        norm_data = fluorescence(:,c)./mean_data; %divide entire data set by
mean to normalize
        sd = nanstd(norm_data); %standard deviation of normalized data
        max = nanmax(norm_data); %find maximum of normalized data
        assessment = (sd./max).*10; %divide standard deviation by maximum.
        stats(pp,:) = [c sdr_n var_n var_pt var_d mean_n mean_data mean_peaks
SNR SNR_t assessment];
        for B = 1:length(PBINS1);
            %include variance of noise before/after each peak?
            peaknum = [smoothnormdata(:,c) peakindx(:,c) peakindx2(:,c)
peakindx3(:,c)];
            peaktop_m = peaknum(find(peaknum(:,3) == B & peaknum(:,2) ==
2));%peak top data
            peakt_num = peaknum(find(peaknum(:,3) == B & peaknum(:,4) == 3));
%peak data
```

```
            sdr_p = nanstd(peakt_num); %stdev of whole peak (cyan color)
            SNR_p = nanmean(peaktop_m)./sdr_n;
            sdr_ip = nanstd(peaktop_m); %stdev of individual peak top
            var_ip = nanvar(peaktop_m);
            peak_max = nanmax(peaktop_m);
            peak_min = nanmin(peaktop_m);
            peaktop_range = peak_max-peak_min; %range of peak top max/min
            Fv=fval(p,9);
            Fv_a=fval(p,10);
            Fv_w=fval(p,11);
            FvFm_p = fval(p,12);
            FvFm_a = fval(p,13);
            FvFm_w = fval(p,14);
            peakdiff1 = NaN;
            peakdiff_a = NaN;
            peakdiff_w = NaN;
            if ~isempty(peaktop_range)
                if Fv > 0
                    peakdiff1 = (peaktop_range./Fv).*100;
                end
                if Fv_a > 0
                    peakdiff_a = (peaktop_range./Fv_a).*100;
                end
                if Fv_w > 0
                    peakdiff_w = (peaktop_range./Fv_w).*100;
                end
            end
            % percent variation table comparing 2 calc methods to WALZ
output.
            Fo_c = abs((fval(B,3)-fval(B,4))./fval(B,3)).*100;%
            Fo_d1 = abs((fval(B,3)-fval(B,5))./fval(B,3)).*100;
            Fo_d2 = abs((fval(B,4)-fval(B,5))./fval(B,4)).*100;
            Fm_c = abs((fval(B,6)-fval(B,7))./fval(B,6)).*100;
            Fm_d1 = abs((fval(B,6)-fval(B,8))./fval(B,6)).*100;
            Fm_d2 = abs((fval(B,7)-fval(B,8))./fval(B,7)).*100;
            Fv_c = abs((fval(B,9)-fval(B,10))./fval(B,9)).*100;
            Fv_d1 = abs((fval(B,9)-fval(B,11))./fval(B,9)).*100; %point vs W
            Fv_d2 = abs((fval(B,10)-fval(B,11))./fval(B,10)).*100; %avg vs W
            FvFm_c = abs((fval(B,12)-fval(B,13))./fval(B,12)).*100;
            FvFm_d1 = abs((fval(B,12)-fval(B,14))./fval(B,12)).*100;
            FvFm_d2 = abs((fval(B,13)-fval(B,14))./fval(B,13)).*100;
            if B == 1
                fvals1(pp,:) = [c Fv Fv_a Fv_w FvFm_p FvFm_a FvFm_w];
            end
            peakstats(p,:) = [c B sdr_ip var_ip peakdiff1 peakdiff_a
peakdiff_w];
            percomp(p,:) = [c B Fo_c Fo_d1 Fo_d2 Fm_c Fm_d1 Fm_d2 Fv_c Fv_d1
Fv_d2 FvFm_c FvFm_d1 FvFm_d2 sdr_p SNR_p];

            %---Comparing Recovery & Percent Variation of Recovery
%Percent CV of 1st to last FvFm
            prval = [fval(PBINS1(1),12) fval(PBINS1(13),12)];
            stdevpr = nanstd(prval);
            avgpr = nanmean(prval);
            FvFm_pr = (stdevpr./avgpr).*100;
            arval = [fval(PBINS1(1),13) fval(PBINS1(13),13)];
            stdevar = nanstd(arval);
```

```
            avgar = nanmean(arval);
            FvFm_ar = (stdevar./avgar).*100;
            wrval = [fval(PBINS1(1),14) fval(PBINS1(13),14)];
            stdevwr = nanstd(wrval);
            avgwr = nanmean(wrval);
            FvFm_wr = (stdevwr./avgwr).*100;
            perrecov(pp,:) = [c FvFm_pr FvFm_ar FvFm_wr];
            p=p+1;
        end
        PBINS1 = (13*c)+1:(13*(c+1));
        pp = pp+1;
    end
    close all
    if option == 1
        save([name ext '_st.mat'],'bfline',
'stats','fvals1','peakstats','percomp','perrecov') %sn for normalized and s
for smoothed
    end
elseif normal == 0
    matname = [name '_sfp.mat']; %need index peak/noise info
    Fname = getfname(matname)
    load([FPpamdir Fname]);
    data = smoothdata;
    ext = '_s';
        pp = 1;
    p = 1;
    PBINS1 = [1:13];
    for c = 1:length(fluorescence(1,:))
        indx = [fluorescence(:,c) peakindx(:,c)];
        noise = indx(find(indx(:,2) ==0)); %noise
        peaktop = indx(find(indx(:,2) ==1)); %top of peak
        chart = chartindx(:,c);
        if chart == 20
            noiseindx = [fluorescence(1:6150,c) peakindx(1:6150,c)];
            noise_vals = noiseindx(find(noiseindx(:,2) == 0));
            x = 1:length(noise_vals);
            x1 = x';
            poly = polyfit(x1,noise_vals,1); %Best fit line, calc slope
            f = polyval(poly,x1); %evaluate equation and gives values
            bfline(c,:) = [c poly(1,1) poly(1,2)];
            figure(c)
            plot(x,noise_vals,'-',x1,f,'o')
            ylabel('Relative Fluorescence', 'FontSize',14)
            title('Baseline Noise (peaks removed)')
            legend('noise', 'best fit line')
            if option == 1
                set(gcf, 'PaperPosition',[0.25 2.5 8 6]);
                print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir name ext '_Fo_'
num2str(c) '.png']);
                set(gcf, 'PaperPosition',[0.25 2.5 4 3]);
                print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir name ext '_Fo_'
num2str(c) 'th.png']);
            end
            close all
        else
            disp('Check to see if chart values are correct')
        end
```

```
        %--------------- Noise/Peak STDEV and VAR calculations---------
        sdr_n = nanstd(noise); %standard deviation
        allpeaks = [smoothnormdata(:,c) peakindx(:,c)];
        peaks_idx = allpeaks(find(peakindx(:,c) == 2));
        var_n = nanvar(noise); %variance
        var_pt = nanvar(peaktop); %not quite right....
        var_d = abs((var_n-var_pt)/var_n).*100;
        mean_n = nanmean(noise);
        %for overall data assessment
        mean_data = nanmean(fluorescence(:,c)); %take mean
        mean_peaks = nanmean(peaks_idx(:,1));
        SNR = mean_data./sdr_n; %signal to noise ratio
        SNR_t = mean_peaks./sdr_n;%with peak tops as average
        norm_data = fluorescence(:,c)./mean_data; %divide entire data set by
mean to normalize
        sd = nanstd(norm_data); %standard deviation of normalized data
        max = nanmax(norm_data); %find maximum of normalized data
        assessment = (sd./max).*10; %divide standard deviation by maximum.
        stats(pp,:) = [c sdr_n var_n var_pt var_d mean_n mean_data mean_peaks
SNR SNR_t assessment];
        for B = 1:length(PBINS1);
            %include variance of noise before/after each peak?
            peaknum = [smoothnormdata(:,c) peakindx(:,c) peakindx2(:,c)];
            peaktop_m = peaknum(find(peaknum(:,3) == B & peaknum(:,2) ==
2));%peak top data
            sdr_ip = nanstd(peaktop_m); %stdev of individual peak top
            var_ip = nanvar(peaktop_m);
            peak_max = nanmax(peaktop_m);
            peak_min = nanmin(peaktop_m);
            peaktop_range = peak_max-peak_min; %range of peak top max/min
            Fv=fval(p,9);
            Fv_a=fval(p,10);
            Fv_w=fval(p,11);
            FvFm_p = fval(p,12);
            FvFm_a = fval(p,13);
            FvFm_w = fval(p,14);
            peakdiff1 = NaN;
            peakdiff_a = NaN;
            peakdiff_w = NaN;
            if ~isempty(peaktop_range)
                if Fv > 0
                    peakdiff1 = (peaktop_range./Fv).*100;
                end
                if Fv_a > 0
                    peakdiff_a = (peaktop_range./Fv_a).*100;
                end
                if Fv_w > 0
                    peakdiff_w = (peaktop_range./Fv_w).*100;
                end
            end
            %percent variation table comparing 2 calc methods to WALZ output.
            Fo_c = abs((fval(B,3)-fval(B,4))./fval(B,3)).*100;
            Fo_d1 = abs((fval(B,3)-fval(B,5))./fval(B,3)).*100;
            Fo_d2 = abs((fval(B,4)-fval(B,5))./fval(B,4)).*100;
            Fm_c = abs((fval(B,6)-fval(B,7))./fval(B,6)).*100;
            Fm_d1 = abs((fval(B,6)-fval(B,8))./fval(B,6)).*100;
            Fm_d2 = abs((fval(B,7)-fval(B,8))./fval(B,7)).*100;
```

```
            Fv_c = abs((fval(B,9)-fval(B,10))./fval(B,9)).*100;
            Fv_d1 = abs((fval(B,9)-fval(B,11))./fval(B,9)).*100; %point vs W
            Fv_d2 = abs((fval(B,10)-fval(B,11))./fval(B,10)).*100; %avg vs W
            FvFm_c = abs((fval(B,12)-fval(B,13))./fval(B,12)).*100;
            FvFm_d1 = abs((fval(B,12)-fval(B,14))./fval(B,12)).*100;
            FvFm_d2 = abs((fval(B,13)-fval(B,14))./fval(B,13)).*100;
            if B == 1
                fvals1(pp,:) = [c Fv Fv_a Fv_w FvFm_p FvFm_a FvFm_w];
            end
            peakstats(p,:) = [c B sdr_ip var_ip peakdiff1 peakdiff_a
peakdiff_w];
            percomp(p,:) = [c B Fo_c Fo_d1 Fo_d2 Fm_c Fm_d1 Fm_d2 Fv_c Fv_d1
Fv_d2 FvFm_c FvFm_d1 FvFm_d2];
            %---Comparing Recovery & Percent Variation of Recovery
            %Percent variation of Recovery of 1st to last FvFm
            if fval(PBINS1(1),12) < 0 | fval(PBINS(13),12) <0
                fval(PBINS1(1),12) = 0;
                fval(PBINS(13),12) = 0;
            end
            if fval(PBINS1(1),13) < 0 | fval(PBINS(13),13) <0
                fval(PBINS1(1),13) = 0;
                fval(PBINS(13),13) = 0;
            end
            if fval(PBINS1(1),14) < 0 | fval(PBINS(13),14) <0
                fval(PBINS1(1),14) = 0;
                fval(PBINS(13),14) = 0;
            end
            FvFm_pr = (fval(PBINS1(1),12)-
fval(PBINS1(13),12))./fval(PBINS1(1),12).*100; %one point
            FvFm_ar = (fval(PBINS1(1),13)-
fval(PBINS1(13),13))./fval(PBINS1(1),13).*100; %avg values
            FvFm_wr = (fval(PBINS1(1),14)-
fval(PBINS1(13),14))./fval(PBINS1(1),14).*100; %WALZ values
            %percent variation of recovery between calculations
            FvFm_cr = abs((FvFm_pr - FvFm_ar)/FvFm_pr).*100;
            FvFm_dr1 = abs((FvFm_pr - FvFm_wr)/FvFm_pr).*100;
            FvFm_dr2 = abs((FvFm_ar - FvFm_wr)/FvFm_ar).*100;
            perrecov(pp,:) = [c FvFm_pr FvFm_ar FvFm_wr FvFm_cr FvFm_dr1
FvFm_dr2];
            p=p+1;
        end
        PBINS1 = (13*c)+1:(13*(c+1));
        pp = pp+1;
    end
    close all
    if option == 1
        save([name ext '_st.mat'],'bfline',
'stats','fvals1','peakstats','percomp','perrecov')
end
else
    disp('You did not enter a valid code')
end
end
```

pamcompare_.m

```
function pamcompare_(name,normal,option);
```

```
%pamcompare_.m
%This function compares PAM's original output from WALZ WinControl to plot
%smoothing or normalizing+smoothing peak/chart calculations (Fv, FvFm &
%ETR) and compare to this programs equivalent calculations. Uses data
%calculations from find_peaks_fnc_.m (requires name_fp.mat or name_sfp.mat
%files to exist).
%Use As: pamcompare_(name,normal,option)
%Input:  name   =   Filename '12UT8' as example, leave off any extensions.
%        normal = 1 Reads in normalized + smoothed name_fp.mat file
%               = 2 Reads in smoothed only name_sfp.mat file
%        option = 1 Saves all the Fv, FvFm, ETR plot comparisons in
%                   HTML_data directory as png figures.
%Output: Figures    Saves Fv,FvFm, ETR figures (.png) in HTML_data dir.
%Created: 16 OCT 2013 by N.Bobco
close all
sysvars_
if nargin == 1
    option = 0;
end
if ~exist(HTMLpamdir)
    mkdir(pwd,HTMLpamdir)
end
PBINS1 = [1:13];
peaks = [1:13];
p = 1;
if normal == 1;
    matname = [name '_fp.mat'];
    Fname = getfname(matname)
    load([FPpamdir Fname])
    ext = '_ns';
    for c = 1:length(chartindx(1,:))
        FvFm_r = fval(PBINS1,14); %r for raw WinControl data
        FvFm_p = fval(PBINS1,12); %p for one point calculation
        FvFm_a = fval(PBINS1,13);%a for average calculation
        figure
        plot(peaks,FvFm_r, '-*r')
        hold on
        plot(FvFm_p, '-*g')
        plot(FvFm_a, '-*b')
        xlabel('peak number', 'FontSize',14)
        ylabel('Fv/Fm', 'FontSize',14)
        title('Comparison of WinControl to Matlab')
        legend('WinControl', 'Single Point', 'Average')
        if option == 1
            set(gcf, 'PaperPosition',[0.25 2.5 8 6])
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir name ext '_FvFm_'
num2str(c) '.png']);
            set(gcf, 'PaperPosition',[0.25 2.5 4 3])
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir name ext '_FvFm_'
num2str(c) 'th' '.png']);
        end
        close all
        Fv_r = fval(PBINS1,11);
        Fv_p = fval(PBINS1,9);
        Fv_a = fval(PBINS1,10);
        figure
        plot(peaks,Fv_r, '-*r')
```

```
        hold on
        plot(Fv_p, '-*g')
        plot(Fv_a, '-*b')
        xlabel('peak number', 'FontSize',14)
        ylabel('Fv', 'FontSize',14)
        title('Comparison of WinControl to Matlab')
        legend('WinControl', 'Single Point', 'Average')
        if option == 1
            set(gcf, 'PaperPosition',[0.25 2.5 8 6])
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir name ext '_Fv_'
num2str(c) '.png']);
            set(gcf, 'PaperPosition',[0.25 2.5 4 3])
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir name ext '_Fv_'
num2str(c) 'th' '.png']);
        end
        close all
        ETR_r = lc(PBINS1,8);
        ETR_p = lc(PBINS1,6);
        ETR_a = lc(PBINS1,7);
        figure
        plot(peaks,ETR_r, '-*r')
        hold on
        plot(ETR_p, '-*g')
        plot(ETR_a, '-*b')
        xlabel('peak number', 'FontSize',14)
        ylabel('ETR', 'FontSize',14)
        title('Comparison of WinControl to Matlab')
        legend('WinControl', 'Single Point', 'Average')
        if option == 1
            set(gcf, 'PaperPosition',[0.25 2.5 8 6])
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir name ext '_ETR_'
num2str(c) '.png']);
            set(gcf, 'PaperPosition',[0.25 2.5 4 3])
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir name ext '_ETR_'
num2str(c) 'th' '.png']);
        end
        close all
    end
    close all
elseif normal == 0;
    matname = [name '_sfp.mat'];
    Fname = getfname(matname)
    load([FPpamdir Fname])
    ext = '_s';
    for c = 1:length(chartindx(1,:))
        FvFm_r = fval(PBINS1,14); %r for raw WinControl data
        FvFm_p = fval(PBINS1,12); %p for one point calculation
        FvFm_a = fval(PBINS1,13);%a for average calculation
        figure
        plot(peaks,FvFm_r, '-*r')
        hold on
        plot(FvFm_p, '-*g')
        plot(FvFm_a, '-*b')
        xlabel('peak number', 'FontSize',14)
        ylabel('Fv/Fm', 'FontSize',14)
        title('Comparison of WinControl to Matlab')
        legend('WinControl', 'Single Point', 'Average')
```

```
        if option == 1
            set(gcf, 'PaperPosition',[0.25 2.5 8 6])
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir name ext '_FvFm_'
num2str(c) '.png']);
            set(gcf, 'PaperPosition',[0.25 2.5 4 3])
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir name ext '_FvFm_'
num2str(c) 'th' '.png']);
        end
        Fv_r = fval(PBINS1,11);
        Fv_p = fval(PBINS1,9);
        Fv_a = fval(PBINS1,10);
        figure
        plot(peaks,Fv_r, '-*r')
        hold on
        plot(Fv_p, '-*g')
        plot(Fv_a, '-*b')
        xlabel('peak number', 'FontSize',14)
        ylabel('Fv', 'FontSize',14)
        title('Comparison of WinControl to Matlab')
        legend('WinControl', 'Single Point', 'Average')
        if option == 1
            set(gcf, 'PaperPosition',[0.25 2.5 8 6])
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir name ext '_Fv_'
num2str(c) '.png']);
            set(gcf, 'PaperPosition',[0.25 2.5 4 3])
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir name ext '_Fv_'
num2str(c) 'th' '.png']);
        end
        ETR_r = lc(PBINS1,8);
        ETR_p = lc(PBINS1,6);
        ETR_a = lc(PBINS1,7);
        figure
        plot(peaks,ETR_r, '-*r')
        hold on
        plot(ETR_p, '-*g')
        plot(ETR_a, '-*b')
        xlabel('peak number', 'FontSize',14)
        ylabel('ETR', 'FontSize',14)
        title('Comparison of WinControl to Matlab')
        legend('WinControl', 'Single Point', 'Average')
        if option == 1
            set(gcf, 'PaperPosition',[0.25 2.5 8 6])
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir name ext '_ETR_'
num2str(c) '.png']);
            set(gcf, 'PaperPosition',[0.25 2.5 4 3])
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir name ext '_ETR_'
num2str(c) 'th' '.png']);
        end
        PBINS1 = (13*c)+1:(13*(c+1));
    end
    close all
else
    disp('You did not a valid code. 1 or 0 only')
end
```

cyclecompare_.m

```
function [pindx_u,pindx_t] =
cyclecompare_(uptake,treatment,number,normal,option);
%cyclecompare_.m
%The purpose of this function is to compares Uptake to Treatment chart
%files by calculating % differences in stats (STD, VAR) for data set as a
%whole as well as for individual peaks.  Also creates peak by peak
%comparison subplots with gray bars showing where peaks should occur
%allowing for visual comparison.
%Use As: cyclecompare_('filename1',filename2',3,1,1);
%Inputs: uptake    = filename of uptake charts (no extensions, '12UT8').
%        treatment = filename of treatment charts (no extensions, '12UT8').
%        number =   Number of charts to be compared.
%        normal = 1 Use normalized/smoothed data file.
%        normal = 0 Use smoothed only data file.
%        option = 1 Save file outputs in current dir. figs in HTML_data dir
%        option = 0 Output (graphs, calculations), not saved.
%
%Prompted Inputs (in program):
%        choice =   Compare uptake chart 1 to treatment chart # (goes in
%                   sequential order for uptake charts (1-number of
comparisons).
%Outputs:
%        Figures    Saves plot comparisons of uptake and treatment chart
(full)
%                   and individual peak comparison plots in HTML_data dir.
%        FILE       Saves new file: uptakevtreatment_cns.mat for norm+smoot
%                   Saves new file: uptakevtreatment_cs.mat for smooth only
%Created: 5 NOV 2013 by N.Bobco
close all
sysvars_
if nargin == 1
    option = 0;
end
p = 1;
if normal == 1;
    matname = [uptake '_ns_fp.mat'];
    Fname = getfname(matname)
    load([FPpamdir Fname])
    matname = [uptake '_ns_st.mat'];
    Fname = getfname(matname)
    load([FPpamdir Fname])
    eval([ 'smoothnormdata' '_u' ' = smoothnormdata;' ]);
    eval([ 'peakindx' '_u' ' = peakindx;' ]);
    eval([ 'peakindx2' '_u' ' = peakindx2;' ]);
    eval([ 'chartindx' '_u' ' = chartindx;' ]);
    eval([ 'fval' '_u' ' = fval;' ]);
    eval([ 'bfline' '_u' ' = bfline;' ]);
    eval([ 'lc' '_u' ' = lc;' ]);
    eval([ 'peakstats' '_u' ' = peakstats;' ]);
    eval([ 'stats' '_u' ' = stats;' ]);
    matname = [treatment '_ns_fp.mat'];
    Fname = getfname(matname)
    load([FPpamdir Fname])
    matname = [treatment '_ns_st.mat'];
    Fname = getfname(matname)
    load([FPpamdir Fname])
```

```
    total = [1:1:number];
    chart = 1
    for c = 1:length(total)
        choice = input('For this uptake chart number, write numerical
treatment chart number for comparison: ');
        u_data = smoothnormdata_u(:,c);
        t_data = smoothnormdata(:,choice);
        u = (1:length(u_data))';
        t = (1:length(t_data))';
        %--Peak Comparisons---%
        if chartindx_u(:,c) == 10
            figure(c)
            subplot(2,1,1)
            multicolorplot(u_data,u_data,220:12150)
            ylabel('Relative Fluorescence')
            title('Control','FontSize',12)
            set(gca,'XLim',[0 24000]) %set max value for 8 min chart length
            set(gca,'XTick',[0:3000:30000]) %about 3000 points per minute.
            set(gca,'XTickLabel',['0';'1';'2';'3';'4';'5';'6';'7';'8'])
            subplot(2,1,2)
            multicolorplot(t_data,t_data,220:12150)
            xlabel('Time (min)')
            ylabel('Relative Fluorescence')
            title('Treatment')
            set(gca,'XLim',[0 24000]) %set max value for 8 min chart length
            set(gca,'XTick',[0:3000:30000]) %about 3000 points per minute.
            set(gca,'XTickLabel',['0';'1';'2';'3';'4';'5';'6';'7';'8'])
            if option == 1
                set(gcf, 'PaperPosition',[0.25 2.5 8 6])
                print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir uptake  '_v_'
treatment '_' num2str(c) '.png']);
            end
            PBINS1 = [  1 1400 2900 4450 5900 7450 8950 10450 12000 12500
14000 17000 23050]; %can make more narrow
            PBINS2 = [220 1700 3200 4750 6200 7750 9250 10750 12250 12800
14300 17300 23350]; %can make more narrow
        elseif chartindx_u(:,c) == 15
            figure(c)
            subplot(2,1,1)
            multicolorplot(u_data,u_data,120:8200)
            ylabel('Relative Fluorescence')
            title('Control','FontSize',12)
            set(gca,'XLim',[0 16000]) %set max value for 8 min chart length
            set(gca,'XTick',[0:2000:18000]) %about 2100 points per minute.
            set(gca,'XTickLabel',['0';'1';'2';'3';'4';'5';'6';'7';'8'])
            subplot(2,1,2)
            multicolorplot(t_data,t_data,120:8200)
            xlabel('Time (min)')
            ylabel('Relative Fluorescence')
            title('Treatment')
            set(gca,'XLim',[0 16000]) %set max value for 8 min chart length
            set(gca,'XTick',[0:2000:18000]) %about 2100 points per minute.
            set(gca,'XTickLabel',['0';'1';'2';'3';'4';'5';'6';'7';'8'])
            PBINS1 = [  1 1000 1950 3000 4000 5000 6000 7020 8050 8450 9425
11450 15500];
            PBINS2 = [120 1120 2150 3200 4200 5200 6200 7250 8220 8560 9600
11650 15700];
```

```
        elseif chartindx_u(:,c) == 20
            figure(c)
            subplot(2,1,1)
            multicolorplot(u_data,u_data,100:6150)
            ylabel('Relative Fluorescence')
            title('Control', 'FontSize',12)
            set(gca,'XLim',[0 13000]) %set max value for 8 min chart length
            set(gca,'XTick',[0:1500:30000]) %about 1500 points per minute.
            set(gca,'XTickLabel',['0';'1';'2';'3';'4';'5';'6';'7';'8'])
            subplot(2,1,2)
            multicolorplot(t_data,t_data,100:6150)
            xlabel('Time (min)')
            ylabel('Relative Fluorescence')
            title('Treatment')
            set(gca,'XLim',[0 13000]) %set max value for 8 min chart length
            set(gca,'XTick',[0:1500:30000]) %about 1500 points per minute.
            set(gca,'XTickLabel',['0';'1';'2';'3';'4';'5';'6';'7';'8'])

            if option == 1
                set(gcf, 'PaperPosition',[0.25 2.5 8 6])
                print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir uptake  'v'
treatment '_ns_' num2str(c) '.png']);
            end
            close all
            pindx_u =[u u_data peakindx2_u(:,c) peakindx_u(:,c)];
            pindx_t =[t t_data peakindx2(:,c) peakindx(:,c)];
            for B = 1:13
                peak_u = u_data(find(pindx_u(:,3) == B ));
                peak_t = t_data(find(pindx_t(:,3) == B ));
                u_x = pindx_u(find(pindx_u(:,3) == B ));
                t_x = pindx_t(find(pindx_t(:,3) == B ));
                top = pindx_u(find(pindx_u(:,3) == B & pindx_u(:,4) == 2));
                %optional, use top of treatment peak for second gray bar
                top_t = pindx_t(find(pindx_t(:,3) == B & pindx_t(:,4) == 2));
                max_u = nanmax(u_data(top))+50;
                min_u = nanmin(peak_u);
                min_t = nanmin(peak_t);
                if ~isempty(top_t)
                    max_t = nanmax(t_data(top_t))+30;
                else
                    max_t = min_t.*3;
                end
                first = top(1)-10;
                last = top(end)+10;
                %--------- Overlay Figure -------------%
                figure(B)
                subplot(2,1,1)
                plot(u_x,peak_u,'g','LineWidth', 1.5) %green line control
                ph = patch([first first last last],[min_u max_u max_u
min_u],[-10 -10 -10 -10],[0.5 0.5 0.5]);
                set(ph, 'EdgeColor','none')
                ylabel('RFU')
                title(['Peak ',num2str(B),': Control'],'FontSize',12)
                subplot(2,1,2)
                plot(t_x,peak_t,'r','LineWidth', 1.5)               ph =
patch([first first last last],[min_t max_t max_t min_t],[-10 -10 -10 -
10],[0.5 0.5 0.5]);
```

```
                set(ph, 'EdgeColor','none')
                ylabel('RFU')
                title('Treatment')
                if option == 1
                    set(gcf, 'PaperPosition',[0.25 2.5 8 6])
                    print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir uptake 'v'
treatment '_c' num2str(c) '_p' num2str(B) '.png']);
                    set(gcf, 'PaperPosition',[0.25 2.5 4 3])
                    print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir uptake 'v'
treatment '_c' num2str(c) '_p' num2str(B) 'th.png']);
                end
                FvFm_c = fval_u(p,14); %average calc value for control.
                FvFm_t = fval(p,14);
                SNR_c = fval_u(p,15);
                SNR_t = fval(p,15);
                slope_c = bfline_u(i,2);
                slope_t = bfline(i,2);
                score_c = stats_u(i,11);
                score_t = stats(i,11);
                p_comp(p,:)=[c B FvFm_c FvFm_t SNR_c SNR_t slope_c slope_t
score_c score_t];
                 close all
                p = p+1;
            end
            chart = chart + 1
        else
            disp('Charts must be 20 min in length')
        end
        if option == 1
            save([uptake 'v' treatment
'_cns.mat'],'p_comp','stats_u','stats','total');
        end
    end
    elseif normal == 0
    matname = [uptake '_sfp.mat'];
    Fname = getfname(matname)
    load([FPpamdir Fname])
    matname = [uptake '_s_st.mat'];
    Fname = getfname(matname)
    load([FPpamdir Fname])
    matname = [treatment '_sfp.mat'];
    Fname = getfname(matname)
    load([FPpamdir Fname])
    matname = [treatment '_s_st.mat'];
    Fname = getfname(matname)
    load([FPpamdir Fname])
else
    disp('Incorrect input: 1 or 0 only')
end
end
```

mkhtmlpg_.m

```
function mkhtmlpg_(name,type,normal,option);
%mkhtmlpg.m
%This function's purpose is to make html pages from the saved pngs and
%data calculations for easy visualization on one page.
```

```
%Use As: mkhtmlpg_('filename',type,normal,option);
%Input:  name  =    filename ex/ '13UTD1.pcf' original name. Use as '13UTD1'.
Auto
%                   finds the extra extension. For type 2, use full new
%                   name of created comparison file ex/ 'DRD1017vDRD1022'.
%        type   = 1  For regular html pages for individual chart files
%        type   = 2  For cyclecompare_.m comparison of uptake v treatment
%        normal= 1  For normalized/smoothed data
%        normal= 0  For smoothed only data
%        option= 1  save .mat file and print figs to html_pam directory
%        option= 0  does not save file to directory.
%Output: FILE       Html page with figures and tables in HTML_data dir.
%Created: 16 OCT 2013 by N.Bobco
sysvars_
if nargin == 1
    option = 0;
end
if ~exist(HTMLpamdir)
    mkdir(pwd,HTMLpamdir)
end
%-----------Single chart html page maker------------%
if type == 1;
    PBINS1 = [1:13];
    if normal == 1;
        matname = [name '_fp.mat'];
        Fname = getfname(matname)
        load([FPpamdir Fname])
        matname = [name '_ns_st.mat'];
        Fname = getfname(matname)
        load([FPpamdir Fname])
        ext = '_ns';
        if option == 1
            for c = 1:length(chartindx)
                chart = chartindx(c);
                HTMLfile = [HTMLpamdir name ext '_' num2str(c) '.html'];
                clear htmlS
                R = 1;
                htmlS{R} = {['<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">']};R = R +1;
                htmlS{R} = {['<html
xmlns="http://www.w3.org/1999/xhtml">']};R = R +1;
                htmlS{R} = {['<head>']};R = R +1;
                htmlS{R} = {['<meta http-equiv="Content-Type"
content="text/html; charset=utf-8" />']};R = R +1;
                htmlS{R} = {['<title>' name '</title>']};R = R +1;
                htmlS{R} = {['<style type="text/css">']};R = R +1;
                htmlS{R} = {['<!--']};R = R +1;
                htmlS{R} = {['.style1 {']};R = R +1;
                htmlS{R} = {['font-size: larger;']};R = R +1;
                htmlS{R} = {['font-weight: bold;']};R = R +1;
                htmlS{R} = {['}']};R = R +1;
                htmlS{R} = {['-->']};R = R +1;
                htmlS{R} = {['</style>']};R = R +1;
                htmlS{R} = {['</head>']};R = R +1;
                htmlS{R} = {['<body><p>REV date: ' datestr(now) '</p>']};R =
R +1;
```

```
                htmlS{R} = {['']};R = R +1;
                htmlS{R} = {['<center><p class="style1">' name
'</p></center>']};R = R +1;
                htmlS{R} = {['<p>PWD: ' pwd '</p>']};R = R +1;
                htmlS{R} = {['<p>PWD: ' Fname '</p>']};R = R +1;
                htmlS{R} = {['<p>PWD: ' mfilename '</p>']};R = R +1;
                htmlS{R} = {['<p>']};R = R +1;
                htmlS{R} = {['PAM light curve data plots ']};R = R +1;
                htmlS{R} = {['  ']};R = R +1;
                htmlS{R} = {['  ']};R = R +1;
                htmlS{R} = {['  ']};R = R +1;
                htmlS{R} = {['</p>']};R = R +1;
                %------ Raw, Norm, Norm & Smoothed Chart Plots ------%
                D = dir([HTMLpamdir name ext '_r' num2str(c) '.png']);
%adding raw chart at top
                if ~isempty(D)
                    for k = 1:size(D,1)
                        htmlS{R} = {['<td <center><img src = "'
deblank(lower(D(k).name)) '" ></center></td>']};R = R +1;
                    end
                end
                httmlS{R} = {['<hr/>']};R = R +1;
                htmlS{R} = {['<table border="0" width="95%"> ']};R = R +1;
                htmlS{R} = {['<tr>']};R = R +1;
                htmlS{R} = {['<th align="center">Normalized</th>']};R = R +1;
                htmlS{R} = {['<th align="center">Normalized and
Smoothed<_a/th>']};R = R +1;
                htmlS{R} = {['<tr>']};R = R +1;
                D = dir([HTMLpamdir name ext '_' num2str(chart) 'n*'
num2str(c) '.png']); %adding normalized and smoothed charts
                if ~isempty(D)
                    htmlS{R} = {['<td <center><a href="' [name ext '_'
num2str(chart) 'n_' num2str(c) '.png'] '"><img src = "' [name ext '_'
num2str(chart) 'n_' num2str(c) 'th.png'] '" ></a><center></td>']};R = R +1;
                    htmlS{R} = {['<td <center><a href="' [name ext '_'
num2str(chart) 'ns_' num2str(c) '.png'] '"><img src = "' [name ext '_'
num2str(chart) 'ns_' num2str(c) 'th.png'] '" ></a><center></td>']};R = R +1;
                end
                htmlS{R} = {['</table>']};R = R +1;
                htmlS{R} = {['<hr/']};R = R +1;
                htmlS{R} = {['</hr>']};R = R +1;
                %---- Table for Fvalues ----------%
                httmlS{R} = {['<hr/>']};R = R +1;
                htmlS{R} = {['<table border="1" width="95%"> ']};R = R +1;
                htmlS{R} = {['<tr>']};R = R +1;
                htmlS{R} = {['<th align="center">Peak number</th>']};R = R
+1;
                htmlS{R} = {['<th align="center">Fo</th>']};R = R +1;
                htmlS{R} = {['<th align="center">Fo_a</th>']};R = R +1;
                htmlS{R} = {['<th align="center">Fo_w</th>']};R = R +1;
                htmlS{R} = {['<th align="center">Fm</th>']};R = R +1;
                htmlS{R} = {['<th align="center">Fm_a</th>']};R = R +1;
                htmlS{R} = {['<th align="center">Fm_w</th>']};R = R +1;
                htmlS{R} = {['<th align="center">Fv</th>']};R = R +1;
                htmlS{R} = {['<th align="center">Fv_a</th>']};R = R +1;
                htmlS{R} = {['<th align="center">Fv_w</th>']};R = R +1;
                htmlS{R} = {['<th align="center">Fv/Fm</th>']};R = R +1;
```

```
                htmlS{R} = {['<th align="center">Fv/Fm_a<_a/th>']};R = R +1;
                htmlS{R} = {['<th align="center">Fv/Fm_w<_a/th>']};R = R +1;
                htmlS{R} = {['<tr>']};R = R +1;
                for B = 1:length(PBINS1)
                    htmlS{R} = {['<tr>']};R = R +1; %may or may not need
                    if B >=2 & B <= 9
                        htmlS{R} = {['<td align="center" bgcolor="#FF0000">'
num2str(fval(B,2)) '</td>']};R = R +1; %num2str(fval(B,2))
                    else
                        htmlS{R} = {['<td align="center"
bgcolor="#0000FF"><font color="#FFFFFF">' num2str(fval(B,2)) '</td>']};R = R
+1;
                    end
                    htmlS{R} = {['<td align="center">'
num2str(fval(PBINS1(B),3)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(fval(PBINS1(B),4)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(fval(PBINS1(B),5)) '</td>']};R = R +1; %get orig data b/c not all
save properly
                    htmlS{R} = {['<td align="center">'
num2str(fval(PBINS1(B),6)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(fval(PBINS1(B),7)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(fval(PBINS1(B),8)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(fval(PBINS1(B),9)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(fval(PBINS1(B),10)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(fval(PBINS1(B),11)) '</td>']};R = R +1; %figure out
                    htmlS{R} = {['<td align="center">'
num2str(fval(PBINS1(B),12)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(fval(PBINS1(B),13)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(fval(PBINS1(B),14),'%8.4f') '</td>']};R = R +1;
                end
                htmlS{R} = {['</table>']};R = R +1;
                PBINS1 = (13*c)+1:(13*(c+1)); %FOR CORRECT TABLE INFO
                D = dir([HTMLpamdir name ext '_' num2str(chart) '*.png']);
                if ~isempty(D)
                    htmlS{R} = {['<td <center><img src = "' [name ext '_'
num2str(chart) '_c' num2str(c) '.png'] '" ></center></td>']};R = R +1;
                end
                %--- Individual Peak Plots -----%
                D = dir([HTMLpamdir name ext '_' num2str(chart) '_c'
num2str(c) '*.png']);
                if ~isempty(D)
                    httmlS{R} = {['<hr/>']};R = R +1;
                    htmlS{R} = {['<table border="0" width="95%"> ']};R = R
+1;
                    htmlS{R} = {['<tr>']};R = R +1;
                    htmlS{R} = {['<td <center><a href="' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '1.png'] '"><img src = "' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '1th.png'] '" ></a><center></td>']};R = R
```

```
+1;
                        htmlS{R} = {['<td <center><a href="' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '2.png'] '"><img src = "' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '2th.png'] '" ></a><center></td>']};R = R
+1;
                        htmlS{R} = {['<td <center><a href="' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '3.png'] '"><img src = "' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '3th.png'] '" ></a><center></td>']};R = R
+1;
                        htmlS{R} = {['<td <center><a href="' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '4.png'] '"><img src = "' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '4th.png'] '" ></a><center></td>']};R = R
+1;
                        htmlS{R} = {['<tr>']};R = R +1;
                        htmlS{R} = {['<td <center><a href="' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '5.png'] '"><img src = "' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '5th.png'] '" ></a><center></td>']};R = R
+1;
                        htmlS{R} = {['<td <center><a href="' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '6.png'] '"><img src = "' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '6th.png'] '" ></a><center></td>']};R = R
+1;
                        htmlS{R} = {['<td <center><a href="' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '7.png'] '"><img src = "' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '7th.png'] '" ></a><center></td>']};R = R
+1;
                        htmlS{R} = {['<td <center><a href="' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '8.png'] '"><img src = "' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '8th.png'] '" ></a><center></td>']};R = R
+1;
                        htmlS{R} = {['<tr>']};R = R +1;
                        htmlS{R} = {['<td <center><a href="' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '9.png'] '"><img src = "' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '9th.png'] '" ></a><center></td>']};R = R
+1;
                        htmlS{R} = {['<td <center><a href="' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '10.png'] '"><img src = "' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '10th.png'] '" ></a><center></td>']};R =
R +1;
                        htmlS{R} = {['<td <center><a href="' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '11.png'] '"><img src = "' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '11th.png'] '" ></a><center></td>']};R =
R +1;
                        htmlS{R} = {['<td <center><a href="' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '12.png'] '"><img src = "' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '12th.png'] '" ></a><center></td>']};R =
R +1;
                        htmlS{R} = {['<tr>']};R = R +1;
                        htmlS{R} = {['<td <center><a href="' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '13.png'] '"><img src = "' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '13th.png'] '" ></a><center></td>']};R =
R +1;
                        htmlS{R} = {['<td <center> ' ' <center></td>']};R = R +1;
                        htmlS{R} = {['<td <center> ' ' <center></td>']};R = R +1;
                        htmlS{R} = {['<tr>']};R = R +1;
                        htmlS{R} = {['</table>']};R = R +1;
                end
```

```
                htmlS{R} = {['<hr/']};R = R +1;
                htmlS{R} = {['</hr>']};R = R +1;
                %----- Light Curve Calculations: ETR, qP... --------%
                PBINS1 = [1:13];
                httmlS{R} = {['<hr/>']};R = R +1;
                htmlS{R} = {['<table border="1" width="95%"> ']};R = R +1;
                htmlS{R} = {['<tr>']};R = R +1;
                htmlS{R} = {['<th align="center">Peak number</th>']};R = R
+1;
                htmlS{R} = {['<th align="center">qP_w</th>']};R = R +1;
                htmlS{R} = {['<th align="center">NPQ_w</th>']};R = R +1;
                htmlS{R} = {['<th align="center">PAR</th>']};R = R +1;
                htmlS{R} = {['<th align="center">ETR_p</th>']};R = R +1;
                htmlS{R} = {['<th align="center">ETR_a</th>']};R = R +1;
                htmlS{R} = {['<th align="center">ETR_w<_a/th>']};R = R +1;
                htmlS{R} = {['<tr>']};R = R +1;
                for B = 1:length(PBINS1)
                    htmlS{R} = {['<tr>']};R = R +1; %may or may not need
                    if B >=2 & B <= 9
                        htmlS{R} = {['<td align="center" bgcolor="#FF0000">'
num2str(fval(B,2)) '</td>']};R = R +1; %num2str(fval(B,2))
                    else
                        htmlS{R} = {['<td align="center"
bgcolor="#0000FF"><font color="#FFFFFF">' num2str(fval(B,2)) '</td>']};R = R
+1;
                    end
                    htmlS{R} = {['<td align="center">'
num2str(lc(PBINS1(B),3)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(lc(PBINS1(B),4)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(lc(PBINS1(B),5)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(lc(PBINS1(B),6)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(lc(PBINS1(B),7)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(lc(PBINS1(B),8),'%8.4f') '</td>']};R = R +1;
                end
                htmlS{R} = {['</table>']};R = R +1;
                PBINS1 = (13*c)+1:(13*(c+1));
                D = dir([HTMLpamdir name ext '*.png']);
                if ~isempty(D)
                    httmlS{R} = {['<hr/>']};R = R +1;
                    htmlS{R} = {['<table border="0" width="95%"> ']};R = R
+1;
                    htmlS{R} = {['<td <center><a href="' [name ext '_FvFm_'
num2str(c) '.png'] '"><img src = "' [name ext '_FvFm_' num2str(c) 'th.png']
'" ></a><center></td>']};R = R +1;
                    htmlS{R} = {['<td <center><a href="' [name ext '_Fv_'
num2str(c) '.png'] '"><img src = "' [name ext '_Fv_'  num2str(c) 'th.png'] '"
></a><center></td>']};R = R +1;
                    htmlS{R} = {['<tr>']};R = R +1;
                    htmlS{R} = {['<td <center><a href="' [name ext '_Fo_'
num2str(c) '.png'] '"><img src = "' [name ext '_Fo_' num2str(c) 'th.png'] '"
></a><center></td>']};R = R +1;
```

```
                    htmlS{R} = {['<td <center><a href="' [name ext '_ETR_'
num2str(c) '.png'] '"><img src = "' [name ext '_ETR_'  num2str(c) 'th.png']
'" ></a><center></td>']};R = R +1;
                    htmlS{R} = {['<tr>']};R = R +1;
                    htmlS{R} = {['</table>']};R = R +1;
                end
                %----- Stat comparisons -------%
                PBINS1 = [1:13];
                httmlS{R} = {['<hr/>']};R = R +1;
                htmlS{R} = {['<table border="1" width="95%"> ']};R = R +1;
                htmlS{R} = {['<tr>']};R = R +1;
                htmlS{R} = {['<th align="center">FvFm 1:13 compare</th>']};R
= R +1;
                htmlS{R} = {['<th align="center">% recovery Point
Calc</th>']};R = R +1;
                htmlS{R} = {['<th align="center">% recovery Avg
Calc</th>']};R = R +1;
                htmlS{R} = {['<th align="center">% recovery WALZ</th>']};R =
R +1;
                htmlS{R} = {['<th align="center">% Diff P:A</th>']};R = R +1;
                htmlS{R} = {['<th align="center">% Diff P:W</th>']};R = R +1;
                htmlS{R} = {['<th align="center">% Diff A:W<_a/th>']};R = R
+1;
                htmlS{R} = {['<tr>']};R = R +1;
                htmlS{R} = {['<td align="center">' ' ' '</td>']};R = R +1;
%num2str(fval(B,2))
                htmlS{R} = {['<td align="center">' num2str(perrecov(c,2))
'</td>']};R = R +1;
                htmlS{R} = {['<td align="center">' num2str(perrecov(c,3))
'</td>']};R = R +1;
                htmlS{R} = {['<td align="center">' num2str(perrecov(c,4))
'</td>']};R = R +1;
                htmlS{R} = {['<td align="center">' num2str(perrecov(c,5))
'</td>']};R = R +1;
                htmlS{R} = {['<td align="center">' num2str(perrecov(c,6))
'</td>']};R = R +1;
                htmlS{R} = {['<td align="center">'
num2str(perrecov(c,7),'%8.4f') '</td>']};R = R +1;
                htmlS{R} = {['</table>']};R = R +1;
                htmlS{R} = {['<hr/']};R = R +1;
                htmlS{R} = {['</hr>']};R = R +1;
                httmlS{R} = {['<hr/>']};R = R +1;
                htmlS{R} = {['<table border="1" width="95%"> ']};R = R +1;
                htmlS{R} = {['<tr>']};R = R +1;
                htmlS{R} = {['<th align="center">Peak number</th>']};R = R
+1;
                htmlS{R} = {['<th align="center">% Diff Fo P:A</th>']};R = R
+1;
                htmlS{R} = {['<th align="center">% Diff Fo P:W</th>']};R = R
+1;
                htmlS{R} = {['<th align="center">% Diff Fo A:W</th>']};R = R
+1;
                htmlS{R} = {['<th align="center">% Diff Fm P:A</th>']};R = R
+1;
                htmlS{R} = {['<th align="center">% Diff Fm P:W</th>']};R = R
+1;
                htmlS{R} = {['<th align="center">% Diff Fm A:W</th>']};R = R
```

```
+1;
                htmlS{R} = {['<th align="center">% Diff Fv P:W</th>']};R = R
+1;
                htmlS{R} = {['<th align="center">% Diff Fv A:W</th>']};R = R
+1;
                htmlS{R} = {['<th align="center">% Diff Fv P:A</th>']};R = R
+1;
                htmlS{R} = {['<th align="center">% Diff FvFm P:W</th>']};R =
R +1;
                htmlS{R} = {['<th align="center">% Diff FvFm A:W</th>']};R =
R +1;
                htmlS{R} = {['<th align="center">% Diff FvFm A:W<_a/th>']};R
= R +1;
                htmlS{R} = {['<tr>']};R = R +1;
                for B = 1:length(PBINS1)
                    htmlS{R} = {['<tr>']};R = R +1; %may or may not need
                    if B >=2 & B <= 9
                        htmlS{R} = {['<td align="center" bgcolor="#FF0000">'
num2str(percomp(PBINS1(B),2)) '</td>']};R = R +1; %num2str(fval(B,2))
                    else
                        htmlS{R} = {['<td align="center"
bgcolor="#0000FF"><font color="#FFFFFF">' num2str(percomp(PBINS1(B),2))
'</td>']};R = R +1;
                    end
                    htmlS{R} = {['<td align="center">'
num2str(percomp(PBINS1(B),3)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(percomp(PBINS1(B),4)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(percomp(PBINS1(B),5)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(percomp(PBINS1(B),6)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(percomp(PBINS1(B),7)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(percomp(PBINS1(B),8)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(percomp(PBINS1(B),9)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(percomp(PBINS1(B),10)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(percomp(PBINS1(B),11)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(percomp(PBINS1(B),12)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(percomp(PBINS1(B),13)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(percomp(PBINS1(B),14),'%8.4f') '</td>']};R = R +1;
                end
                htmlS{R} = {['</table>']};R = R +1;
                PBINS1 = (13*c)+1:(13*(c+1));
                %------------------- Stats output ---------------------%
                htmlS{R} = {['<hr/']};R = R +1;
                htmlS{R} = {['</hr>']};R = R +1;
                httmlS{R} = {['<hr/>']};R = R +1;
                htmlS{R} = {['<table border="1" width="95%"> ']};R = R +1;
                htmlS{R} = {['<tr>']};R = R +1;
```

```
                htmlS{R} = {['<th align="center">Stats</th>']};R = R +1;
                htmlS{R} = {['<th align="center">Var. Noise</th>']};R = R +1;
                htmlS{R} = {['<th align="center">Var. Peak Tops</th>']};R = R
+1;
                htmlS{R} = {['<th align="center">STDEV Noise</th>']};R = R
+1;
                htmlS{R} = {['<th align="center">mean noise</th>']};R = R +1;
                htmlS{R} = {['<th align="center">mean data</th>']};R = R +1;
                htmlS{R} = {['<th align="center">mean peak top</th>']};R = R
+1;
                htmlS{R} = {['<th align="center">SNR</th>']};R = R +1;
                htmlS{R} = {['<th align="center">score<_a/th>']};R = R +1;
                htmlS{R} = {['<tr>']};R = R +1;
                htmlS{R} = {['<td align="center">' ' ' '</td>']};R = R +1;
                htmlS{R} = {['<td align="center">' num2str(stats(c,2))
'</td>']};R = R +1;
                htmlS{R} = {['<td align="center">' num2str(stats(c,3))
'</td>']};R = R +1;
                htmlS{R} = {['<td align="center">' num2str(stats(c,4))
'</td>']};R = R +1;
                htmlS{R} = {['<td align="center">' num2str(stats(c,5))
'</td>']};R = R +1;
                htmlS{R} = {['<td align="center">' num2str(stats(c,6))
'</td>']};R = R +1;
                htmlS{R} = {['<td align="center">' num2str(stats(c,7))
'</td>']};R = R +1;
                htmlS{R} = {['<td align="center">' num2str(stats(c,9))
'</td>']};R = R +1;
                htmlS{R} = {['<td align="center">'
num2str(stats(c,11),'%8.4f') '</td>']};R = R +1;
                htmlS{R} = {['</table>']};R = R +1;
                htmlS{R} = {['<hr/']};R = R +1;
                htmlS{R} = {['</hr>']};R = R +1;
                %------End of tables and graphs--------%
                htmlS{R} = {['<br/>']};R = R +1;
                htmlS{R} = {['<p>REV date: ' datestr(now) '</p>']};R = R +1;
                htmlS{R} = {['</body>']};R = R +1;
                htmlS{R} = {['</html>']};R = R +1;
                fid = fopen(HTMLfile,'wt');
                if fid > 0
                    [M N] = size(htmlS);
                    for S = 1:N
                        fwrite(fid,[char(htmlS{S}) 10],'char');
                    end
                    fclose(fid);;
                else
                    disp([FCNname ' Error (Cound not open' HTMLfile ')'])
                end
                disp(['  Created ' HTMLfile])
            end
        elseif normal == 0;
            matname = [name '_sfp.mat'];
            Fname = getfname(matname)
            ext = '_s';
            if option == 1
                for c = 1:length(chartindx)
                    chart = chartindx(c);
```

```
                    HTMLfile = [HTMLpamdir name ext '_' num2str(c) '.html'];
                    clear htmlS
                    R = 1;
                    htmlS{R} = {['<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML
1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">']};R = R +1;
                    htmlS{R} = {['<html
xmlns="http://www.w3.org/1999/xhtml">']};R = R +1;
                    htmlS{R} = {['<head>']};R = R +1;
                    htmlS{R} = {['<meta http-equiv="Content-Type"
content="text/html; charset=utf-8" />']};R = R +1;
                    htmlS{R} = {['<title>' name '</title>']};R = R +1;
                    htmlS{R} = {['<style type="text/css">']};R = R +1;
                    htmlS{R} = {['<!--']};R = R +1;
                    htmlS{R} = {['.style1 {']};R = R +1;
                    htmlS{R} = {['font-size: larger;']};R = R +1;
                    htmlS{R} = {['font-weight: bold;']};R = R +1;
                    htmlS{R} = {['}']};R = R +1;
                    htmlS{R} = {['-->']};R = R +1;
                    htmlS{R} = {['</style>']};R = R +1;
                    htmlS{R} = {['</head>']};R = R +1;
                    htmlS{R} = {['<body><p>REV date: ' datestr(now)
'</p>']};R = R +1;
                    htmlS{R} = {['']};R = R +1;
                    htmlS{R} = {['<center><p class="style1">' name
'</p></center>']};R = R +1;

                    htmlS{R} = {['<p>PWD: ' pwd '</p>']};R = R +1;
                    htmlS{R} = {['<p>PWD: ' Fname '</p>']};R = R +1;
                    htmlS{R} = {['<p>PWD: ' mfilename '</p>']};R = R +1;
                    htmlS{R} = {['<p>']};R = R +1;
                    htmlS{R} = {['PAM light curve data plots ']};R = R +1;
                    htmlS{R} = {['  ']};R = R +1;
                    htmlS{R} = {['  ']};R = R +1;
                    htmlS{R} = {['  ']};R = R +1;
                    htmlS{R} = {['</p>']};R = R +1;
                    %------ Raw, Norm, Norm & Smoothed Chart Plots ------%
                    D = dir([HTMLpamdir name ext '_r' num2str(c) '.png']);
%adding raw chart at top
                    if ~isempty(D)
                        for k = 1:size(D,1)
                            htmlS{R} = {['<td <center><img src = "'
deblank(lower(D(k).name)) '" ></center></td>']};R = R +1;
                        end
                    end
                    httmlS{R} = {['<hr/>']};R = R +1;
                    htmlS{R} = {['<table border="0" width="95%"> ']};R = R
+1;

                    htmlS{R} = {['<tr>']};R = R +1;
                    htmlS{R} = {['<th align="center">Smoothed<_a/th>']};R = R
+1;

                    htmlS{R} = {['<tr>']};R = R +1;
                    D = dir([HTMLpamdir name ext '_' num2str(chart) 'rs*'
num2str(c) '.png']); %adding normalized and smoothed charts
                    if ~isempty(D)
```

```
                    htmlS{R} = {['<td <center><a href="' [name ext '_'
num2str(chart) 'rs_' num2str(c) '.png'] '"><img src = "' [name ext '_'
num2str(chart) 'rs_' num2str(c) 'th.png'] '" ></a><center></td>']};R = R +1;
                end
                htmlS{R} = {['</table>']};R = R +1;
                htmlS{R} = {['<hr/']};R = R +1;
                htmlS{R} = {['</hr>']};R = R +1;
                %---- Table for Fvalues ----------%
                httmlS{R} = {['<hr/>']};R = R +1;
                htmlS{R} = {['<table border="1" width="95%"> ']};R = R
+1;
                htmlS{R} = {['<tr>']};R = R +1;
                htmlS{R} = {['<th align="center">Peak number</th>']};R =
R +1;
                htmlS{R} = {['<th align="center">Fo</th>']};R = R +1;
                htmlS{R} = {['<th align="center">Fo_a</th>']};R = R +1;
                htmlS{R} = {['<th align="center">Fo_w</th>']};R = R +1;
                htmlS{R} = {['<th align="center">Fm</th>']};R = R +1;
                htmlS{R} = {['<th align="center">Fm_a</th>']};R = R +1;
                htmlS{R} = {['<th align="center">Fm_w</th>']};R = R +1;
                htmlS{R} = {['<th align="center">Fv</th>']};R = R +1;
                htmlS{R} = {['<th align="center">Fv_a</th>']};R = R +1;
                htmlS{R} = {['<th align="center">Fv_w</th>']};R = R +1;
                htmlS{R} = {['<th align="center">Fv/Fm</th>']};R = R +1;
                htmlS{R} = {['<th align="center">Fv/Fm_a<_a/th>']};R = R
+1;
                htmlS{R} = {['<th align="center">Fv/Fm_w<_a/th>']};R = R
+1;
                htmlS{R} = {['<tr>']};R = R +1;
                for B = 1:length(PBINS1)
                    htmlS{R} = {['<tr>']};R = R +1; %may or may not need
                    if B >=2 & B <= 9
                        htmlS{R} = {['<td align="center"
bgcolor="#FF0000">' num2str(fval(B,2)) '</td>']};R = R +1;
%num2str(fval(B,2))
                    else
                        htmlS{R} = {['<td align="center"
bgcolor="#0000FF"><font color="#FFFFFF">' num2str(fval(B,2)) '</td>']};R = R
+1;
                    end
                    htmlS{R} = {['<td align="center">'
num2str(fval(PBINS1(B),3)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(fval(PBINS1(B),4)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(fval(PBINS1(B),5)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(fval(PBINS1(B),6)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(fval(PBINS1(B),7)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(fval(PBINS1(B),8)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(fval(PBINS1(B),9)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(fval(PBINS1(B),10)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
```

```
num2str(fval(PBINS1(B),11)) '</td>']};R = R +1;
                        htmlS{R} = {['<td align="center">'
num2str(fval(PBINS1(B),12)) '</td>']};R = R +1;
                        htmlS{R} = {['<td align="center">'
num2str(fval(PBINS1(B),13)) '</td>']};R = R +1;
                        htmlS{R} = {['<td align="center">'
num2str(fval(PBINS1(B),14),'%8.4f') '</td>']};R = R +1;
                    end
                    htmlS{R} = {['</table>']};R = R +1;
                    PBINS1 = (13*c)+1:(13*(c+1)); %FOR CORRECT TABLE INFO
                    D = dir([HTMLpamdir name ext '_' num2str(chart)
'*.png']);
                    if ~isempty(D)
                        htmlS{R} = {['<td <center><img src = "' [name '_'
num2str(chart) '_c' num2str(c) '.png'] '" ></center></td>']};R = R +1;
                    end
                    %---- Individual Peak Plots ---------%
                    D = dir([HTMLpamdir name ext '_' num2str(chart) '_c'
num2str(c) '*.png']);
                    if ~isempty(D)
                        httmlS{R} = {['<hr/>']};R = R +1;
                        htmlS{R} = {['<table border="0" width="95%"> ']};R =
R +1;
                        htmlS{R} = {['<tr>']};R = R +1;
                        htmlS{R} = {['<td <center><a href="' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '1.png'] '"><img src = "' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '1th.png'] '" ></a><center></td>']};R = R
+1;
                        htmlS{R} = {['<td <center><a href="' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '2.png'] '"><img src = "' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '2th.png'] '" ></a><center></td>']};R = R
+1;
                        htmlS{R} = {['<td <center><a href="' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '3.png'] '"><img src = "' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '3th.png'] '" ></a><center></td>']};R = R
+1;
                        htmlS{R} = {['<td <center><a href="' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '4.png'] '"><img src = "' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '4th.png'] '" ></a><center></td>']};R = R
+1;
                        htmlS{R} = {['<tr>']};R = R +1;
                        htmlS{R} = {['<td <center><a href="' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '5.png'] '"><img src = "' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '5th.png'] '" ></a><center></td>']};R = R
+1;
                        htmlS{R} = {['<td <center><a href="' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '6.png'] '"><img src = "' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '6th.png'] '" ></a><center></td>']};R = R
+1;
                        htmlS{R} = {['<td <center><a href="' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '7.png'] '"><img src = "' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '7th.png'] '" ></a><center></td>']};R = R
+1;
                        htmlS{R} = {['<td <center><a href="' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '8.png'] '"><img src = "' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '8th.png'] '" ></a><center></td>']};R = R
+1;
```

```
                    htmlS{R} = {['<tr>']};R = R +1;
                    htmlS{R} = {['<td <center><a href="' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '9.png'] '"><img src = "' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '9th.png'] '" ></a><center></td>']};R = R
+1;
                    htmlS{R} = {['<td <center><a href="' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '10.png'] '"><img src = "' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '10th.png'] '" ></a><center></td>']};R =
R +1;
                    htmlS{R} = {['<td <center><a href="' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '11.png'] '"><img src = "' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '11th.png'] '" ></a><center></td>']};R =
R +1;
                    htmlS{R} = {['<td <center><a href="' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '12.png'] '"><img src = "' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '12th.png'] '" ></a><center></td>']};R =
R +1;
                    htmlS{R} = {['<tr>']};R = R +1;
                    htmlS{R} = {['<td <center><a href="' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '13.png'] '"><img src = "' [name ext '_'
num2str(chart) '_c' num2str(c) '_p' '13th.png'] '" ></a><center></td>']};R =
R +1;
                    htmlS{R} = {['<td <center> ' ' <center></td>']};R = R
+1;
                    htmlS{R} = {['<td <center> ' ' <center></td>']};R = R
+1;
                    htmlS{R} = {['<tr>']};R = R +1;
                    htmlS{R} = {['</table>']};R = R +1;
                end
                htmlS{R} = {['<hr/']};R = R +1;
                htmlS{R} = {['</hr>']};R = R +1;
                %------Light Curve Calculations: ETR, qP... ---------%
                PBINS1 = [1:13];
                httmlS{R} = {['<hr/>']};R = R +1;
                htmlS{R} = {['<table border="1" width="95%"> ']};R = R
+1;
                htmlS{R} = {['<tr>']};R = R +1;
                htmlS{R} = {['<th align="center">Peak number</th>']};R =
R +1;
                htmlS{R} = {['<th align="center">qP_w</th>']};R = R +1;
                htmlS{R} = {['<th align="center">NPQ_w</th>']};R = R +1;
                htmlS{R} = {['<th align="center">PAR</th>']};R = R +1;
                htmlS{R} = {['<th align="center">ETR_p</th>']};R = R +1;
                htmlS{R} = {['<th align="center">ETR_a</th>']};R = R +1;
                htmlS{R} = {['<th align="center">ETR_w<_a/th>']};R = R
+1;
                htmlS{R} = {['<tr>']};R = R +1;
                for B = 1:length(PBINS1)
                    htmlS{R} = {['<tr>']};R = R +1; %may or may not need
                    if B >=2 & B <= 9
                        htmlS{R} = {['<td align="center"
bgcolor="#FF0000">' num2str(lc(PBINS1(B),2)) '</td>']};R = R +1;
%num2str(fval(B,2))
                    else
                        htmlS{R} = {['<td align="center"
bgcolor="#0000FF"><font color="#FFFFFF">' num2str(lc(PBINS1(B),2))
'</td>']};R = R +1;
```

```
                        end
                        htmlS{R} = {['<td align="center">'
num2str(lc(PBINS1(B),3)) '</td>']};R = R +1;
                        htmlS{R} = {['<td align="center">'
num2str(lc(PBINS1(B),4)) '</td>']};R = R +1;
                        htmlS{R} = {['<td align="center">'
num2str(lc(PBINS1(B),5)) '</td>']};R = R +1;
                        htmlS{R} = {['<td align="center">'
num2str(lc(PBINS1(B),6)) '</td>']};R = R +1;
                        htmlS{R} = {['<td align="center">'
num2str(lc(PBINS1(B),7)) '</td>']};R = R +1;
                        htmlS{R} = {['<td align="center">'
num2str(lc(PBINS1(B),8),'%8.4f') '</td>']};R = R +1;
                    end
                    htmlS{R} = {['</table>']};R = R +1;
                    PBINS1 = (13*c)+1:(13*(c+1));
                    D = dir([HTMLpamdir name ext '*.png']);
                    if ~isempty(D)
                        httmlS{R} = {['<hr/>']};R = R +1;
                        htmlS{R} = {['<table border="0" width="95%"> ']};R =
R +1;
                        htmlS{R} = {['<td <center><a href="' [name ext
'_FvFm_' num2str(c) '.png'] '"><img src = "' [name ext '_FvFm_' num2str(c)
'th.png'] '" ></a><center></td>']};R = R +1;
                        htmlS{R} = {['<td <center><a href="' [name ext '_Fv_'
num2str(c) '.png'] '"><img src = "' [name ext '_Fv_'  num2str(c) 'th.png'] '"
></a><center></td>']};R = R +1;
                        htmlS{R} = {['<tr>']};R = R +1;
                        htmlS{R} = {['<td <center><a href="' [name ext '_Fo_'
num2str(c) '.png'] '"><img src = "' [name ext '_Fo_' num2str(c) 'th.png'] '"
></a><center></td>']};R = R +1;
                        htmlS{R} = {['<td <center><a href="' [name ext
'_ETR_' num2str(c) '.png'] '"><img src = "' [name ext '_ETR_'  num2str(c)
'th.png'] '" ></a><center></td>']};R = R +1;
                        htmlS{R} = {['<tr>']};R = R +1;
                        htmlS{R} = {['</table>']};R = R +1;
                    end
                    htmlS{R} = {['<hr/']};R = R +1;
                    htmlS{R} = {['</hr>']};R = R +1;
                    %----- Stat comparisons -------%
                    PBINS1 = [1:13];
                    httmlS{R} = {['<hr/>']};R = R +1;
                    htmlS{R} = {['<table border="1" width="95%"> ']};R = R
+1;
                    htmlS{R} = {['<tr>']};R = R +1;
                    htmlS{R} = {['<th align="center">FvFm 1:13
compare</th>']};R = R +1;
                    htmlS{R} = {['<th align="center">% recovery Point
Calc</th>']};R = R +1;
                    htmlS{R} = {['<th align="center">% recovery Avg
Calc</th>']};R = R +1;
                    htmlS{R} = {['<th align="center">% recovery
WALZ</th>']};R = R +1;
                    htmlS{R} = {['<th align="center">% Diff P:A</th>']};R = R
+1;
                    htmlS{R} = {['<th align="center">% Diff P:W</th>']};R = R
+1;
```

```
                    htmlS{R} = {['<th align="center">% Diff A:W<_a/th>']};R =
R +1;
                    htmlS{R} = {['<tr>']};R = R +1;
                    htmlS{R} = {['<td align="center">' num2str(perrecov(c,2))
'</td>']};R = R +1; %num2str(fval(B,2))
                    htmlS{R} = {['<td align="center">' num2str(perrecov(c,3))
'</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">' num2str(perrecov(c,4))
'</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">' num2str(perrecov(c,5))
'</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">' num2str(perrecov(c,6))
'</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">' num2str(perrecov(c,7))
'</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(perrecov(c,8),'%8.4f') '</td>']};R = R +1;
                    htmlS{R} = {['</table>']};R = R +1;
                    htmlS{R} = {['<hr/']};R = R +1;
                    htmlS{R} = {['</hr>']};R = R +1;
                    httmlS{R} = {['<hr/>']};R = R +1;
                    htmlS{R} = {['<table border="1" width="95%"> ']};R = R
+1;
                    htmlS{R} = {['<tr>']};R = R +1;
                    htmlS{R} = {['<th align="center">Peak number</th>']};R =
R +1;
                    htmlS{R} = {['<th align="center">% Diff Fo P:A</th>']};R
= R +1;
                    htmlS{R} = {['<th align="center">% Diff Fo P:W</th>']};R
= R +1;
                    htmlS{R} = {['<th align="center">% Diff Fo A:W</th>']};R
= R +1;
                    htmlS{R} = {['<th align="center">% Diff Fm P:A</th>']};R
= R +1;
                    htmlS{R} = {['<th align="center">% Diff Fm P:W</th>']};R
= R +1;
                    htmlS{R} = {['<th align="center">% Diff Fm A:W</th>']};R
= R +1;
                    htmlS{R} = {['<th align="center">% Diff Fv P:W</th>']};R
= R +1;
                    htmlS{R} = {['<th align="center">% Diff Fv A:W</th>']};R
= R +1;
                    htmlS{R} = {['<th align="center">% Diff Fv P:A</th>']};R
= R +1;
                    htmlS{R} = {['<th align="center">% Diff FvFm
P:W</th>']};R = R +1;
                    htmlS{R} = {['<th align="center">% Diff FvFm
A:W</th>']};R = R +1;
                    htmlS{R} = {['<th align="center">% Diff FvFm
A:W<_a/th>']};R = R +1;
                    htmlS{R} = {['<tr>']};R = R +1;
                    for B = 1:length(PBINS1)
                        htmlS{R} = {['<tr>']};R = R +1; %may or may not need
                        if B >=2 & B <= 9
                            htmlS{R} = {['<td align="center"
bgcolor="#FF0000">' num2str(percomp(PBINS1(B),2)) '</td>']};R = R +1;
%num2str(fval(B,2))
```

```
                    else
                        htmlS{R} = {['<td align="center"
bgcolor="#0000FF"><font color="#FFFFFF">' num2str(percomp(PBINS1(B),2))
'</td>']};R = R +1;
                    end
                    htmlS{R} = {['<td align="center">'
num2str(percomp(PBINS1(B),3)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(percomp(PBINS1(B),4)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(percomp(PBINS1(B),5)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(percomp(PBINS1(B),6)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(percomp(PBINS1(B),7)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(percomp(PBINS1(B),8)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(percomp(PBINS1(B),9)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(percomp(PBINS1(B),10)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(percomp(PBINS1(B),11)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(percomp(PBINS1(B),12)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(percomp(PBINS1(B),13)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(percomp(PBINS1(B),14),'%8.4f') '</td>']};R = R +1;
                end
                htmlS{R} = {['</table>']};R = R +1;
                PBINS1 = (13*c)+1:(13*(c+1));
                %-------------------- Stats output -----------------%
                htmlS{R} = {['<hr/']};R = R +1;
                htmlS{R} = {['</hr>']};R = R +1;
                httmlS{R} = {['<hr/>']};R = R +1;
                htmlS{R} = {['<table border="1" width="95%"> ']};R = R
+1;
                htmlS{R} = {['<tr>']};R = R +1;
                htmlS{R} = {['<th align="center">Stats</th>']};R = R +1;
                htmlS{R} = {['<th align="center">Var. Noise</th>']};R = R
+1;
                htmlS{R} = {['<th align="center">Var. Peak Tops</th>']};R
= R +1;
                htmlS{R} = {['<th align="center">STDEV Noise</th>']};R =
R +1;
                htmlS{R} = {['<th align="center">mean noise</th>']};R = R
+1;
                htmlS{R} = {['<th align="center">mean data</th>']};R = R
+1;
                htmlS{R} = {['<th align="center">mean peak top</th>']};R
= R +1;
                htmlS{R} = {['<th align="center">SNR</th>']};R = R +1;
                htmlS{R} = {['<th align="center">score<_a/th>']};R = R
+1;
                htmlS{R} = {['<tr>']};R = R +1;
```

```
                    htmlS{R} = {['<td align="center">' ' ' '</td>']};R = R
+1;
                    htmlS{R} = {['<td align="center">' num2str(stats(c,2))
'</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">' num2str(stats(c,3))
'</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">' num2str(stats(c,4))
'</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">' num2str(stats(c,5))
'</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">' num2str(stats(c,6))
'</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">' num2str(stats(c,7))
'</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">' num2str(stats(c,8))
'</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(stats(c,9),'%8.4f') '</td>']};R = R +1;
                    htmlS{R} = {['</table>']};R = R +1;
                    htmlS{R} = {['<hr/']};R = R +1;
                    htmlS{R} = {['</hr>']};R = R +1;
                    %------End of tables and graphs--------%
                    htmlS{R} = {['<br/>']};R = R +1;
                    htmlS{R} = {['<p>REV date: ' datestr(now) '</p>']};R = R
+1;
                    htmlS{R} = {['</body>']};R = R +1;
                    htmlS{R} = {['</html>']};R = R +1;

                    fid = fopen(HTMLfile,'wt');
                    if fid > 0
                        [M N] = size(htmlS);
                        for S = 1:N
                            fwrite(fid,[char(htmlS{S}) 10],'char');
                        end
                        fclose(fid);;
                    else
                        disp([FCNname ' Error (Cound not open' HTMLfile ')'])
                    end
                    disp(['  Created ' HTMLfile])
                end
            else
                disp('You did not enter a valid code. 1 or 0 only')
            end
        end
    end
    %-----------Pam Cycle Compare html page maker-------------%
elseif type == 2
    if normal == 1
        matname = [name '_cns.mat'];
        Fname = getfname(matname)
        load([FPpamdir Fname])
        ext = '_ns_';
        PBINS1 = [1:13];
        if option == 1
            for c = 1:length(total)
                chart = total(c);
```

```
            HTMLfile = [HTMLpamdir name ext num2str(c) '.html'];
            clear htmlS
            R = 1;
            htmlS{R} = {['<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">']};R = R +1;
            htmlS{R} = {['<html
xmlns="http://www.w3.org/1999/xhtml">']};R = R +1;
            htmlS{R} = {['<head>']};R = R +1;
            htmlS{R} = {['<meta http-equiv="Content-Type"
content="text/html; charset=utf-8" />']};R = R +1;
            htmlS{R} = {['<title>' name '</title>']};R = R +1;
            htmlS{R} = {['<style type="text/css">']};R = R +1;
            htmlS{R} = {['<!--']};R = R +1;
            htmlS{R} = {['.style1 {']};R = R +1;
            htmlS{R} = {['font-size: larger;']};R = R +1;
            htmlS{R} = {['font-weight: bold;']};R = R +1;
            htmlS{R} = {['}']};R = R +1;
            htmlS{R} = {['-->']};R = R +1;
            htmlS{R} = {['</style>']};R = R +1;
            htmlS{R} = {['</head>']};R = R +1;
            htmlS{R} = {['<body><p>REV date: ' datestr(now) '</p>']};R =
R +1;
            htmlS{R} = {['']};R = R +1;
            htmlS{R} = {['<center><p class="style1">' name
'</p></center>']};R = R +1;

            htmlS{R} = {['<p>PWD: ' pwd '</p>']};R = R +1;
            htmlS{R} = {['<p>PWD: ' Fname '</p>']};R = R +1;
            htmlS{R} = {['<p>PWD: ' mfilename '</p>']};R = R +1;
            htmlS{R} = {['<p>']};R = R +1;
            htmlS{R} = {['PAM light curve data plots ']};R = R +1;
            htmlS{R} = {[' ']};R = R +1;
            htmlS{R} = {[' ']};R = R +1;
            htmlS{R} = {[' ']};R = R +1;
            htmlS{R} = {['</p>']};R = R +1;
            %--- Normalized/Smoothed Subplot Overall Comparison ----%
            D = dir([HTMLpamdir name ext num2str(c) '.png']); %adding raw
chart at top
            if ~isempty(D)
                for k = 1:size(D,1)
                    htmlS{R} = {['<td <center><img src = "'
deblank(lower(D(k).name)) '" ></center></td>']};R = R +1;
                end
            end
            %--- Overall Stat Comparison Table ---------------%
            httmlS{R} = {['<hr/>']};R = R +1;
            htmlS{R} = {['<table border="1" width="95%"> ']};R = R +1;
            htmlS{R} = {['<tr>']};R = R +1;
            htmlS{R} = {['<th align="center">Stats</th>']};R = R +1;
            htmlS{R} = {['<th align="center">Var. Noise_c</th>']};R = R
+1;
            htmlS{R} = {['<th align="center">Var. Noise_t</th>']};R = R
+1;
            htmlS{R} = {['<th align="center">Var. Peak Tops_c</th>']};R =
R +1;
            htmlS{R} = {['<th align="center">Var. Peak Tops_t</th>']};R =
```

```
R +1;
                %                   htmlS{R} = {['<th align="center">% Diff.
Var.</th>']};R = R +1;
                htmlS{R} = {['<th align="center">SNR_c</th>']};R = R +1;
                htmlS{R} = {['<th align="center">SNR_t</th>']};R = R +1;
                htmlS{R} = {['<th align="center">SNR with Peak
Mean_c</th>']};R = R +1;
                htmlS{R} = {['<th align="center">SNR with Peak
Mean_c</th>']};R = R +1;
                %                   htmlS{R} = {['<th align="center">% Diff.
STDEV.</th>']};R = R +1;
                htmlS{R} = {['<th align="center">Quality_c<_a/th>']};R = R
+1;
                htmlS{R} = {['<th align="center">Quality_t<_a/th>']};R = R
+1;
                htmlS{R} = {['<tr>']};R = R +1;

                htmlS{R} = {['<td align="center">' ' ' '</td>']};R = R +1;
                htmlS{R} = {['<td align="center">' num2str(stats_u(c,3))
'</td>']};R = R +1;
                htmlS{R} = {['<td align="center">' num2str(stats(c,3))
'</td>']};R = R +1;
                htmlS{R} = {['<td align="center">' num2str(stats_u(c,4))
'</td>']};R = R +1;
                htmlS{R} = {['<td align="center">' num2str(stats(c,4))
'</td>']};R = R +1;
                htmlS{R} = {['<td align="center">' num2str(stats_u(c,9))
'</td>']};R = R +1;
                htmlS{R} = {['<td align="center">' num2str(stats(c,9))
'</td>']};R = R +1;
                htmlS{R} = {['<td align="center">' num2str(stats_u(c,10))
'</td>']};R = R +1;
                htmlS{R} = {['<td align="center">' num2str(stats(c,10))
'</td>']};R = R +1;
                htmlS{R} = {['<td align="center">' num2str(stats_u(c,11))
'</td>']};R = R +1;
                htmlS{R} = {['<td align="center">'
num2str(stats(c,11),'%8.4f') '</td>']};R = R +1;
                htmlS{R} = {['</table>']};R = R +1;
                htmlS{R} = {['<hr/']};R = R +1;
                htmlS{R} = {['</hr>']};R = R +1;
                %--- Normalized/Smoothed Subplot Peak Comparison -------%
                D = dir([HTMLpamdir name '_c' num2str(chart) '_p'  '*.png']);
                if ~isempty(D)
                    httmlS{R} = {['<hr/>']};R = R +1;
                    htmlS{R} = {['<table border="0.5" width="95%"> ']};R = R
+1;

                    htmlS{R} = {['<tr>']};R = R +1;
                    htmlS{R} = {['<td <center><a href="' [name '_c'
num2str(c) '_p' '1.png'] '"><img src = "' [name '_c' num2str(c) '_p'
'1th.png'] '" ></a><center></td>']};R = R +1;
                    htmlS{R} = {['<td <center><a href="' [name '_c'
num2str(c) '_p' '2.png'] '"><img src = "' [name '_c' num2str(c) '_p'
'2th.png'] '" ></a><center></td>']};R = R +1;
                    htmlS{R} = {['<tr>']};R = R +1;
                    htmlS{R} = {['<td <center><a href="' [name '_c'
num2str(c) '_p' '3.png'] '"><img src = "' [name '_c' num2str(c) '_p'
```

```
'3th.png'] '" ></a><center></td>']};R = R +1;
                        htmlS{R} = {['<td <center><a href="' [name '_c'
num2str(c) '_p' '4.png'] '"><img src = "' [name '_c' num2str(c) '_p'
'4th.png'] '" ></a><center></td>']};R = R +1;
                        htmlS{R} = {['<tr>']};R = R +1;
                        htmlS{R} = {['<td <center><a href="' [name '_c'
num2str(c) '_p' '5.png'] '"><img src = "' [name '_c' num2str(c) '_p'
'5th.png'] '" ></a><center></td>']};R = R +1;
                        htmlS{R} = {['<td <center><a href="' [name '_c'
num2str(c) '_p' '6.png'] '"><img src = "' [name '_c' num2str(c) '_p'
'6th.png'] '" ></a><center></td>']};R = R +1;
                        htmlS{R} = {['<tr>']};R = R +1;
                        htmlS{R} = {['<td <center><a href="' [name '_c'
num2str(c) '_p' '7.png'] '"><img src = "' [name '_c' num2str(c) '_p'
'7th.png'] '" ></a><center></td>']};R = R +1;
                        htmlS{R} = {['<td <center><a href="' [name '_c'
num2str(c) '_p' '8.png'] '"><img src = "' [name '_c' num2str(c) '_p'
'8th.png'] '" ></a><center></td>']};R = R +1;
                        htmlS{R} = {['<tr>']};R = R +1;
                        htmlS{R} = {['<td <center><a href="' [name '_c'
num2str(c) '_p' '9.png'] '"><img src = "' [name '_c' num2str(c) '_p'
'9th.png'] '" ></a><center></td>']};R = R +1;
                        htmlS{R} = {['<td <center><a href="' [name '_c'
num2str(c) '_p' '10.png'] '"><img src = "' [name '_c' num2str(c) '_p'
'10th.png'] '" ></a><center></td>']};R = R +1;
                        htmlS{R} = {['<tr>']};R = R +1;
                        htmlS{R} = {['<td <center><a href="' [name '_c'
num2str(c) '_p' '11.png'] '"><img src = "' [name '_c' num2str(c) '_p'
'11th.png'] '" ></a><center></td>']};R = R +1;
                        htmlS{R} = {['<td <center><a href="' [name '_c'
num2str(c) '_p' '12.png'] '"><img src = "' [name '_c' num2str(c) '_p'
'12th.png'] '" ></a><center></td>']};R = R +1;
                        htmlS{R} = {['<tr>']};R = R +1;
                        htmlS{R} = {['<td <center><a href="' [name '_c'
num2str(c) '_p' '13.png'] '"><img src = "' [name '_c' num2str(c) '_p'
'13th.png'] '" ></a><center></td>']};R = R +1;
                        htmlS{R} = {['<td <center> ' ' <center></td>']};R = R +1;
                        %               htmlS{R} = {['<td align="center">'
num2str(fval(B,10),'%8.4f') '</td>']};R = R +1;
                        htmlS{R} = {['<tr>']};R = R +1;
                        htmlS{R} = {['</table>']};R = R +1;
                end
                %-- Peak Comparison Calculations ------%
                htmlS{R} = {['<hr/']};R = R +1;
                htmlS{R} = {['</hr>']};R = R +1;
                httmlS{R} = {['<hr/>']};R = R +1;
                htmlS{R} = {['<table border="1" width="95%"> ']};R = R +1;
                htmlS{R} = {['<tr>']};R = R +1;
                htmlS{R} = {['<th align="center">Peak number</th>']};R = R
+1;
                htmlS{R} = {['<th align="center">FvFm_c</th>']};R = R +1;
                htmlS{R} = {['<th align="center">FvFm_t</th>']};R = R +1;
                htmlS{R} = {['<th align="center">% Diff FvFm c:t</th>']};R =
R +1;
                htmlS{R} = {['<th align="center">std_c</th>']};R = R +1;
                htmlS{R} = {['<th align="center">std_t</th>']};R = R +1;
                htmlS{R} = {['<th align="center">Diff std</th>']};R = R +1;
```

```
                htmlS{R} = {['<th align="center">var_c</th>']};R = R +1;
                htmlS{R} = {['<th align="center">var_t</th>']};R = R +1;
                htmlS{R} = {['<th align="center">% Diff var c:t</th>']};R = R
+1;
                htmlS{R} = {['<th align="center">% peak height_c</th>']};R =
R +1;
                htmlS{R} = {['<th align="center">% peak height_t</th>']};R =
R +1;
                htmlS{R} = {['<tr>']};R = R +1;
                for B = 1:length(PBINS1)
                    htmlS{R} = {['<tr>']};R = R +1; %may or may not need
                    if B >=2 & B <= 9
                        htmlS{R} = {['<td align="center" bgcolor="#FF0000">'
num2str(p_comp(PBINS1(B),2)) '</td>']};R = R +1; %num2str(fval(B,2))
                    else
                        htmlS{R} = {['<td align="center"
bgcolor="#0000FF"><font color="#FFFFFF">' num2str(p_comp(PBINS1(B),2))
'</td>']};R = R +1;
                    end
                    htmlS{R} = {['<td align="center">'
num2str(p_comp(PBINS1(B),3)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(p_comp(PBINS1(B),4)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(p_comp(PBINS1(B),5)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(p_comp(PBINS1(B),6)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(p_comp(PBINS1(B),7)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(p_comp(PBINS1(B),8)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(p_comp(PBINS1(B),9)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(p_comp(PBINS1(B),10)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(p_comp(PBINS1(B),11)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(p_comp(PBINS1(B),12)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(p_comp(PBINS1(B),13),'%8.4f') '</td>']};R = R +1;
                end
                htmlS{R} = {['</table>']};R = R +1;
                PBINS1 = (13*c)+1:(13*(c+1));
                htmlS{R} = {['<hr/']};R = R +1;
                htmlS{R} = {['</hr>']};R = R +1;
                %----- End of tables and graphs ------%
                htmlS{R} = {['<br/>']};R = R +1;
                htmlS{R} = {['<p>REV date: ' datestr(now) '</p>']};R = R +1;
                htmlS{R} = {['</body>']};R = R +1;
                htmlS{R} = {['</html>']};R = R +1;
                fid = fopen(HTMLfile,'wt');
                if fid > 0
                    [M N] = size(htmlS);
                    for S = 1:N
                        fwrite(fid,[char(htmlS{S}) 10],'char');
                    end
```

```
                    fclose(fid);;
                else
                    disp([FCNname ' Error (Cound not open' HTMLfile ')'])
                end
                disp(['  Created ' HTMLfile])
            end
        end
    else
        disp('You did not enter a valid code. 1 or 2 only')
    end
else
    disp('You did not enter a valid code. 1 or 2 only')
end
```

mkopttables_.m

```
function mkopttables_
%Makes a table of all experiments with filenames on each line.
sysvars_
HTMLpamdir
HTMLfile = [HTMLpamdir 'experiment.html'];
clear htmlS
R = 1;
htmlS{R} = {['<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">']};R = R +1;
htmlS{R} = {['<html xmlns="http://www.w3.org/1999/xhtml">']};R = R +1;
htmlS{R} = {['<head>']};R = R +1;
htmlS{R} = {['<meta http-equiv="Content-Type" content="text/html;
charset=utf-8" />']};R = R +1;
htmlS{R} = {['<title>ALL PCF HTML</title>']};R = R +1;
htmlS{R} = {['<style type="text/css">']};R = R +1;
htmlS{R} = {['<!--']};R = R +1;
htmlS{R} = {['.style1 {']};R = R +1;
htmlS{R} = {['font-size: larger;']};R = R +1;
htmlS{R} = {['font-weight: bold;']};R = R +1;
htmlS{R} = {['}']};R = R +1;
htmlS{R} = {['-->']};R = R +1;
htmlS{R} = {['</style>']};R = R +1;
htmlS{R} = {['</head>']};R = R +1;
htmlS{R} = {['<body><p>REV date: ' datestr(now) '</p>']};R = R +1;
htmlS{R} = {['']};R = R +1;
htmlS{R} = {['<center><p class="style1">ALL PCF HTML</p></center>']};R = R
+1;

htmlS{R} = {['<p>PWD: ' pwd '</p>']};R = R +1;
htmlS{R} = {['<p>PWD: ' mfilename '</p>']};R = R +1;
%Table for Fvalues
%fval = [B Fo Fo_a Fm Fm_a Fv Fv_a FvFm FvFm_a f_dif NPQ NPQ_a];
httmlS{R} = {['<hr/>']};R = R +1;
htmlS{R} = {['<table border="1" width="95%"> ']};R = R +1;
htmlS{R} = {['<tr>']};R = R +1;
htmlS{R} = {['<th align="center">Filename</th>']};R = R +1;
htmlS{R} = {['<th align="center">data quality</th>']};R = R +1;
htmlS{R} = {['<tr>']};R = R +1;
f = dir([HTMLpamdir '*.html']);
```

```
for F = 1:size(f,1)
    [PP,NAME] = fileparts(f(F).name)
    fx = dir([NAME '_st.mat'])
    if ~isempty(fx)
        load([NAME '_st.mat'])
        htmlS{R} = {['<tr>']};R = R +1;
        htmlS{R} = {[' <td align="center"><a href="' f(F).name '">' f(F).name
'</a></td>']};R = R +1;
        htmlS{R} = {[' <td align="center">' num2str(stats(1,7)) '</td>']};R =
R +1;
        htmlS{R} = {['<tr>']};R = R +1;
    else
        htmlS{R} = {['<tr>']};R = R +1;
        htmlS{R} = {[' <td align="center"><a href="' f(F).name '">' f(F).name
'</a></td>']};R = R +1;
        htmlS{R} = {[' <td align="center">-</td>']};R = R +1;
        htmlS{R} = {['<tr>']};R = R +1;
    end
end
htmlS{R} = {['</table> ']};R = R +1;
htmlS{R} = {['<br/>']};R = R +1;
htmlS{R} = {['<p>REV date: ' datestr(now) '</p>']};R = R +1;
htmlS{R} = {['</body>']};R = R +1;
htmlS{R} = {['</html>']};R = R +1;
fid = fopen(HTMLfile,'wt');
if fid > 0
    [M N] = size(htmlS);
    for S = 1:N
        fwrite(fid,[char(htmlS{S}) 10],'char');
    end
    fclose(fid);;
else
    disp([FCNname ' Error (Cound not open' HTMLfile ')'])
end
disp(['  Created ' HTMLfile])
end
```

metastats_.m

```
function [all_points,all_FvFm,other,other2]=metastats_(step,normal,option);
%metastats_.m
%This function collects stat data from all experiments currently in
combined_prs directory and characterizes noise (STD, VAR, data quality), and
slope of baseline data (best fit line). It uses data calculations from
stats_.m (requires name_ns_st.mat or name_s_st.mat files to exist) and
creates bar graphs with STE bars, line plots and saves to an html output page
in HTML_data.
%Use As: metastats_(normal,option)
%Input:  step   = 1 Imports the data into one large file.
%        step   = 2 Runs the analysis on the new large data file.
%        step   = 3 Creates html output for analyzed data file.
%        normal = 1 Uses all ns_st.mat files for analysis
%                   (normalized+smoothed data)
%        normal = 0 Uses all s_st.mat files for analysis smoothed only data.
%        option = 1 Saves all stat files and all figures
%Output: Figures    Saves all bar graph figures (.png) in HTML_data dir.
%        Files      Saves meta_ns_ms.mat for norm+smoothed data analysis
```

```
%                    Saves meta_s_ms.mat for smoothed only data analysis
%Created: 12 NOV 2013 by N.Bobco
sysvars_;
if nargin == 1
    option = 0;
end
if ~exist(FPpamdir)
    mkdir(FPpamdir)
end
if step == 1
    if normal == 1
        ext = '_ns'
        wildcard = '*_ns_st.mat'
        disp([pwd])
        f = dir(wildcard)
        p = 1;
        c = 1;
        j = 1;
        for F = 1:size(f,1)
            name = (f(F).name)
            load([FPpamdir name])
            for r = 1:size(stats);
                metas(p,:) = [F stats(r,2) stats(r,3) stats(r,4) stats(r,5)
stats(r,6) stats(r,7) stats(r,8) stats(r,9) stats(r,10) stats(r,11)];
                p = p+1;
            end
            ititle = name(1:end-10);
            for g = 1:size(fvals1);
                metapp(j,:) = [c fvals1(g,2) fvals1(g,3) fvals1(g,4)
fvals1(g,5) fvals1(g,6) fvals1(g,7) perrecov(g,2) perrecov(g,3)
perrecov(g,4)];
                j = j + 1;
            end
            index(c,1) =[{ititle}];
            c = c + 1;
        end
        if option == 1
            save(['meta' ext '_ms.mat'],'metas','index','metapp');
        end
    elseif normal == 0
        ext = '_s'
        wildcard = '*_s_st.mat'
        disp([pwd])
        f = dir(wildcard)
        p = 1;
        c = 1;
        for F = 1:size(f,1)
            name = (f(F).name)
            load([FPpamdir name])
            for r = 1:size(stats);
                type = input('Enter: 1 = uptake, 2 = T0, 3 = T, 4 = MQ, 5 =
FSW: ');
                metas(p,:) = [F stats(r,2) stats(r,3) stats(r,4) stats(r,5)
stats(r,6) stats(r,7) stats(r,8) stats(r,9) stats(r,9) type];
                metap(p,:) = [F peakstats(r,5) peakstats(r,6) peakstats(r,7)
type];
```

```
                p = p+1;
            end
            ititle = name(1:end-10);
            r = size(f,1);
            metapp(c,:) = [c fvals1(:,2) fvals1(:,3) fvals1(:,4) fvals1(:,5)
fvals1(:,6) fvals1(:,7) perrecov(:,2) perrecov(:,3) perrecov(:,4)];
            index(c,1) =[{ititle}];
            c = c+1;
        end
        if option == 1
            save(['meta' ext '_ms.mat'],'metas','metap','metapp','index');
        end
    else
        disp('Not a valid code')
    end
elseif step == 2
    if normal == 1
        close all
        matname = ['meta_ns_ms.mat'];
        Fname = getfname(matname)
        load([FPpamdir Fname]);
        ext = '_ns';
        %calc #2
        %--------------------------%
        %using WALZ FvFM calculation%
        assessment = [metas(:,12)];
        FvFms2 = [metapp(:,7)]; %WALZ FvFm
        indx_u = (assessment(find(metas(:,13) == 1)));
        iu_mean = nanmean(indx_u);
        a = length(indx_u);
        indx_t0 = (assessment(find(metas(:,13) == 2)));
        b = length(indx_t0);
        indx_t = (assessment(find(metas(:,13) ==3)));
        it_mean = nanmean(indx_t);
        c = length(indx_t);
        indx_mq = (assessment(find(metas(:,13) == 4)));
        im_mean = nanmean(indx_mq);
        d = length(indx_mq);
        indx_fw = (assessment(find(metas(:,13) == 5)));
        iw_mean = nanmean(indx_fw);
        e = length(indx_fw);
        FvFm_u = (FvFms2(find(metas(:,13) == 1)))';
        Fu_mean = nanmean(FvFm_u);
        f = length(FvFm_u);
        FvFm_t0 = (FvFms2(find(metas(:,13) == 2)))';
        g = length(FvFm_t0);
        FvFm_t = (FvFms2(find(metas(:,13) == 3)))';
        Ft_mean = nanmean(FvFm_t);
        l = length(FvFm_t);
        FvFm_mq = (FvFms2(find(metas(:,13) == 4)));
        Fm_mean = nanmean(FvFm_mq);
        i = length(FvFm_mq);
        FvFm_fw = (FvFms2(find(metas(:,13) == 5)));
        Fw_mean = nanmean(FvFm_fw);
        j = length(FvFm_fw);
%         Fv_u = (FvFms2(find(metas(:,13) == 1)));
```

```
%         Fv_t0 = (FvFms2(find(metas(:,13) == 2)));
%         Fv_t = (FvFms2(find(metas(:,13) == 3)));
        x_uF = repmat(1,1,f);
        x_t0F= repmat(2,1,g);
        x_tF = repmat(3,1,l);
        x_fF = repmat(4,1,i);
        x_mF = repmat(5,1,j);
%         1*1
%         [h,p,ci,stats] = ttest(indx_u,indx_t)
%         1*2
%         [h,p,ci,stats] = ttest(FvFm_u,FvFm_t)
        1*3
        [h,p,ci,stats] = ttest2(indx_t,indx_mq)
        1*4
        [h,p,ci,stats] = ttest2(FvFm_t,FvFm_mq)
        1*5
        [h,p,ci,stats] = ttest2(indx_t,indx_fw)
        1*6
        [h,p,ci,stats] = ttest2(FvFm_t,FvFm_fw)
        1*7
        [h,p,ci,stats] = ttest2(indx_mq,indx_fw)
        STE_us = nanstd(indx_u)%./sqrt(a) %STE for control score (x axis)
        STE_uF = nanstd(FvFm_u)%./sqrt(f); %STE for control FvFm (y axis)
        STE_ts = nanstd(indx_t)%./sqrt(c); %STE for treatment score (x axis)
        STE_tF = nanstd(FvFm_t)%./sqrt(l); %STE for treatment FvFm (y axis)
        STE_ms = nanstd(indx_mq)%./sqrt(d);
        STE_mF = nanstd(FvFm_mq)%./sqrt(i);
        STE_fs = nanstd(indx_fw)%./sqrt(e);
        STE_fF = nanstd(FvFm_fw)%./sqrt(j);
        x1 = iu_mean;
        x2 = it_mean;
        y1 = Fu_mean;
        y2 = Ft_mean;
        x3 = iw_mean;
        x4 = im_mean
        y3 = Fw_mean;
        y4 = Fm_mean
           %--Scatterplot of FvFm v score by itself----%
        figure
        plot(indx_u,FvFm_u,'.g','MarkerSize',20)
        hold on
        plot(indx_t,FvFm_t,'.r','MarkerSize',20)
        plot(indx_fw,FvFm_fw,'.k','MarkerSize',20)
        plot(indx_mq,FvFm_mq,'.m','MarkerSize',20)
        xlabel('Score','FontSize',14)
        ylabel('Dark Adapted Fv/Fm','FontSize',14)
        legend('Control','Treatment','FSW','MQ','Location','East')
        hold off
        if option == 1
            set(gcf, 'PaperPosition',[0.25 2.5 8 6])
            set(gca, 'FontSize',12)
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir 'meta' ext
'_qualityFvFm3.png']);
            set(gcf, 'PaperPosition',[0.25 2.5 4 3])
            set(gca, 'FontSize',8)
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir 'meta' ext
'_qualityFvFm3_th.png']);
```

```
        end
        %Mean with STE plot
        figure

plot(iu_mean,Fu_mean,'g','Marker','diamond','MarkerSize',10,'MarkerFaceColor'
,'g','MarkerEdgeColor','k')
        hold on
plot(it_mean,Ft_mean,'r','Marker','diamond','MarkerSize',10,'MarkerFaceColor'
,'r','MarkerEdgeColor','k')
plot(iw_mean,Fw_mean,'k','Marker','diamond','MarkerSize',10,'MarkerFaceColor'
,'k','MarkerEdgeColor','k')
plot(im_mean,Fm_mean,'m','Marker','diamond','MarkerSize',10,'MarkerFaceColor'
,'m','MarkerEdgeColor','k')
        hold on
        errorbarxy(x3, y3, STE_fs, STE_fF,{'k' 'k' 'k'})
        hold on
        errorbarxy(x4, y4, STE_ms, STE_mF,{'k' 'k' 'k'})
        hold on
        errorbarxy(x2, y2, STE_ts, STE_tF,{'k' 'k' 'k'})
        hold on
        errorbarxy(x1, y1, STE_us, STE_uF,{'k' 'k' 'k'})
        xlabel('Score','FontSize',14)
        ylabel('Dark Adapted Fv/Fm ','FontSize',14)
        legend('Control', 'Treatment','FSW','MQ','Location','East')
        hold off
        if option == 1
            set(gcf, 'PaperPosition',[0.25 2.5 8 6])
            set(gca, 'FontSize',12)
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir 'meta' ext
'_qualityFvFm4.png']);
            set(gcf, 'PaperPosition',[0.25 2.5 4 3])
            set(gca, 'FontSize',8)
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir 'meta' ext
'_qualityFvFm4_th.png']);
        end
        all_points=[indx_u;indx_t0;indx_t;indx_mq;indx_fw];
        all_FvFm=[FvFm_u FvFm_t]';
        other = [FvFm_t0]';
        other2=[FvFm_mq;FvFm_fw];
        %calc #2 by category
        figure
        categories = {'Control','Treatment','FSW','MQ'};
plot(1,iu_mean,'.g','Marker','diamond','MarkerSize',20,'MarkerFaceColor','g',
'MarkerEdgeColor','k')
        hold on
plot(2,it_mean,'.r','Marker','diamond','MarkerSize',20,'MarkerFaceColor','r',
'MarkerEdgeColor','k')
plot(3,iw_mean,'.k','Marker','diamond','MarkerSize',20,'MarkerFaceColor','k',
'MarkerEdgeColor','k')
plot(4,im_mean,'.m','Marker','diamond','MarkerSize',20,'MarkerFaceColor','m',
'MarkerEdgeColor','k')
        errorbar(1,iu_mean, STE_us)
        errorbar(2,it_mean, STE_ts)
        errorbar(3,iw_mean, STE_fs)
        errorbar(4,im_mean, STE_ms)
        set(gca,'XLim',[0 5])
        set(gca,'XTick',1:4,'XTickLabel',categories)
```

```
        hold off
        if option == 1
            set(gcf, 'PaperPosition',[0.25 2.5 8 6])
            set(gca, 'FontSize',12)
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir 'meta' ext
'_qualityhist.png']);
            set(gcf, 'PaperPosition',[0.25 2.5 4 3])
            set(gca, 'FontSize',8)
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir 'meta' ext
'_qualityhist_th.png']);
        end
        figure
        categories = {'Control','Treatment','FSW','MQ'};
        plot(1,indx_u,'.g','MarkerSize',20)
        hold on
        plot(2,indx_t,'.r','MarkerSize',20)
        plot(3,indx_fw,'.k','MarkerSize',20)
        plot(4,indx_mq,'.m','MarkerSize',20)
        set(gca,'XLim',[0 5])
        set(gca,'XTick',1:4,'XTickLabel',categories)
        hold off
        if option == 1
            set(gcf, 'PaperPosition',[0.25 2.5 8 6])
            set(gca, 'FontSize',12)
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir 'meta' ext
'_qualityhist1.png']);
            set(gcf, 'PaperPosition',[0.25 2.5 4 3])
            set(gca, 'FontSize',8)
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir 'meta' ext
'_qualityhist1_th.png']);
        end
%       %-----Do scatter plot FvFm by category-------%
        figure
        categories = {'Control','T0','Treatment','FSW','MQ'};
        plot(x_uF,FvFm_u,'.g','MarkerSize',20)
        hold on
        plot(x_t0F,FvFm_t0,'.b','MarkerSize',20)
        plot(x_tF,FvFm_t,'.r','MarkerSize',20)
        plot(x_fF,FvFm_fw,'.k','MarkerSize',20)
        plot(x_mF,FvFm_mq,'.m','MarkerSize',20)
        ylabel('Fv/Fm ','FontSize',14)
        set(gca,'XLim',[0 6])
        set(gca,'YLim',[-0.01 0.8])
        set(gca,'XTick',1:5,'XTickLabel',categories)
        hold off
        if option == 1
            set(gcf, 'PaperPosition',[0.25 2.5 8 6])
            set(gca, 'FontSize',12)
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir 'meta' ext
'_assessbox1.png']);
            set(gcf, 'PaperPosition',[0.25 2.5 4 3])
            set(gca, 'FontSize',8)
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir 'meta' ext
'_assessbox1_th.png']);
        end
        figure
        categories = {'Control','T0','Treatment','FSW','MQ'};
```

```
        plot(x_uF,FvFm_u,'.g','MarkerSize',20)
        hold on
        plot(x_t0F,FvFm_t0,'.b','MarkerSize',20)
        plot(x_tF,FvFm_t,'.r','MarkerSize',20)
        plot(x_fF,FvFm_fw,'.k','MarkerSize',20)
        plot(x_mF,FvFm_mq,'.m','MarkerSize',20)
        ylabel('Fv/Fm (initial) ','FontSize',14)
        set(gca,'XLim',[0 6])
        set(gca,'YLim',[-0.01 0.8])
        set(gca,'XTick',1:5,'XTickLabel',categories)
        ph = patch([0 0 6 6],[0.5 0.75 0.75 0.5],[-10 -10 -10 -10]);
        set(ph,'EdgeColor','g','LineWidth',3,'FaceColor','none')
        ph2 = patch([0 0 6 6],[0 0.2 0.2 0],[-10 -10 -10 -10]);
        set(ph2,'EdgeColor','r','LineWidth',3,'FaceColor','none')
        hold off
        if option == 1
            set(gcf, 'PaperPosition',[0.25 2.5 8 6])
            set(gca, 'FontSize',12)
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir 'meta' ext
'_assessbox2.png']);
            set(gcf, 'PaperPosition',[0.25 2.5 4 3])
            set(gca, 'FontSize',8)
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir 'meta' ext
'_assessbox2_th.png']);
        end
        %------- FvFm vs SNR ----------------------------%
        SNRs = [metas(:,10)];
        FvFms = [metapp(:,7)]; %FvFm WALZ

        indx_u = (SNRs(find(metas(:,13) == 1)));
        iu_mean = nanmean(indx_u);
        a = length(indx_u);
        indx_t0 = (SNRs(find(metas(:,13) == 2)));
        b = length(indx_t0);
        indx_t = (SNRs(find(metas(:,13) ==3)));
        it_mean = nanmean(indx_t);
        c = length(indx_t);
        indx_mq = (SNRs(find(metas(:,13) == 4)));
        im_mean = nanmean(indx_mq);
        d = length(indx_mq);
        indx_fw = (SNRs(find(metas(:,13) == 5)));
        iw_mean = nanmean(indx_fw);
        e = length(indx_fw);
        FvFm_u = (FvFms2(find(metas(:,13) == 1)))';
        Fu_mean = nanmean(FvFm_u);
        f = length(FvFm_u);
        FvFm_t0 = (FvFms2(find(metas(:,13) == 2)))';
        g = length(FvFm_t0);
        FvFm_t = (FvFms2(find(metas(:,13) == 3)))';
        Ft_mean = nanmean(FvFm_t);
        l = length(FvFm_t);
        FvFm_mq = (FvFms2(find(metas(:,13) == 4)));
        Fm_mean = nanmean(FvFm_mq);
        i = length(FvFm_mq);
        FvFm_fw = (FvFms2(find(metas(:,13) == 5)));
        Fw_mean = nanmean(FvFm_fw);
        j = length(FvFm_fw);
```

```
        Fv_u = (FvFms(find(metas(:,13) == 1)));
        Fv_t0 = (FvFms(find(metas(:,13) == 2)));
        Fv_t = (FvFms(find(metas(:,13) == 3)));
        x_uF = repmat(1,1,f);
        x_t0F= repmat(2,1,g);
        x_tF = repmat(3,1,l);
        x_mF = repmat(4,1,i);
        x_fF = repmat(5,1,j);
        1*1
        [h,p,ci,stats] = ttest(indx_u,indx_t)
        1*2
        [h,p,ci,stats] = ttest(FvFm_u,FvFm_t)
        1*3
        [h,p,ci,stats] = ttest2(indx_t,indx_mq)
        1*4
        [h,p,ci,stats] = ttest2(FvFm_t,FvFm_mq)
        1*5
        [h,p,ci,stats] = ttest2(indx_t,indx_fw)
        1*6
        [h,p,ci,stats] = ttest2(FvFm_t,FvFm_fw)
        1*7
        [h,p,ci,stats] = ttest2(indx_mq,indx_fw)
        STE_us = nanstd(indx_u)%./sqrt(a) %STE for control score (x axis)
        STE_uF = nanstd(FvFm_u)%./sqrt(f); %STE for control FvFm (y axis)
        STE_ts = nanstd(indx_t)%./sqrt(c); %STE for treatment score (x axis)
        STE_tF = nanstd(FvFm_t)%./sqrt(l); %STE for treatment FvFm (y axis)
        STE_ms = nanstd(indx_mq)%./sqrt(d);
        STE_mF = nanstd(FvFm_mq)%./sqrt(i);
        STE_fs = nanstd(indx_fw)%./sqrt(e);
        STE_fF = nanstd(FvFm_fw)%./sqrt(j);
        x1 = iu_mean;
        x2 = it_mean;
        y1 = Fu_mean;
        y2 = Ft_mean;
        x3 = iw_mean;
        x4 = im_mean;
        y3 = Fw_mean;
        y4 = Fm_mean;
        %--Scatterplot of FvFm v SNR by itself----%
        figure
        plot(indx_u,FvFm_u,'.g','MarkerSize',20)
        hold on
%         plot(indx_t0,FvFm_t0,'.b')
        plot(indx_t,FvFm_t,'.r','MarkerSize',20)
        plot(indx_fw,FvFm_fw,'.k','MarkerSize',20)
        plot(indx_mq,FvFm_mq,'.m','MarkerSize',20)
        xlabel('SNR','FontSize',14)
        ylabel('Fv/Fm (initial) ','FontSize',14)
        legend('Control','Treatment','FSW','MQ','Location','East')
        set(gca,'YLim',[-0.02 0.8])
        hold off
        if option == 1
            set(gcf, 'PaperPosition',[0.25 2.5 8 6])
            set(gca, 'FontSize',12)
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir 'meta' ext
'_SNRFvFm1.png']);
            set(gcf, 'PaperPosition',[0.25 2.5 4 3])
```

```
            set(gca, 'FontSize',8)
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir 'meta' ext
'_SNRFvFm1_th.png']);
        end
        figure
        categories = {'Control','Treatment','FSW','MQ'};
        plot(1,indx_u,'.g','MarkerSize',20)
        hold on
        plot(2,indx_t,'.r','MarkerSize',20)
        plot(3,indx_fw,'.k','MarkerSize',20)
        plot(4,indx_mq,'.m','MarkerSize',20)
        ylabel('SNR','FontSize',14)
        set(gca,'XLim',[0 5])
        set(gca,'XTick',1:4,'XTickLabel',categories)
        hold off
        if option == 1
            set(gcf, 'PaperPosition',[0.25 2.5 8 6])
            set(gca, 'FontSize',12)
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir 'meta' ext
'_SNRhist.png']);
            set(gcf, 'PaperPosition',[0.25 2.5 4 3])
            set(gca, 'FontSize',8)
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir 'meta' ext
'_SNRhist_th.png']);
        end
        figure
        categories = {'Control','Treatment','FSW','MQ'};
plot(1,iu_mean,'.g','Marker','diamond','MarkerSize',20,'MarkerFaceColor','g',
'MarkerEdgeColor','k')
        hold on
plot(2,it_mean,'.r','Marker','diamond','MarkerSize',20,'MarkerFaceColor','r',
'MarkerEdgeColor','k')
plot(3,iw_mean,'.k','Marker','diamond','MarkerSize',20,'MarkerFaceColor','k',
'MarkerEdgeColor','k')
plot(4,im_mean,'.m','Marker','diamond','MarkerSize',20,'MarkerFaceColor','m',
'MarkerEdgeColor','k')
        hold on
        errorbar(1,iu_mean, STE_us)
        errorbar(2,it_mean, STE_ts)
        errorbar(3,iw_mean, STE_fs)
        errorbar(4,im_mean, STE_ms)
        ylabel('Mean SNR','FontSize',14)
        set(gca,'XLim',[0 5])
        set(gca,'XTick',1:4,'XTickLabel',categories)
        hold off
        if option == 1
            set(gcf, 'PaperPosition',[0.25 2.5 8 6])
            set(gca, 'FontSize',12)
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir 'meta' ext
'_SNRhist2.png']);
            set(gcf, 'PaperPosition',[0.25 2.5 4 3])
            set(gca, 'FontSize',8)
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir 'meta' ext
'_SNRhist2_th.png']);
        end
    %Mean with STE plot
        figure
```

```
plot(iu_mean,Fu_mean,'g','Marker','diamond','MarkerSize',10,'MarkerFaceColor'
,'g','MarkerEdgeColor','k')
        hold on
plot(it_mean,Ft_mean,'r','Marker','diamond','MarkerSize',10,'MarkerFaceColor'
,'r','MarkerEdgeColor','k')
plot(iw_mean,Fw_mean,'k','Marker','diamond','MarkerSize',10,'MarkerFaceColor'
,'k','MarkerEdgeColor','k')
plot(im_mean,Fm_mean,'m','Marker','diamond','MarkerSize',10,'MarkerFaceColor'
,'m','MarkerEdgeColor','k')
        hold on
        errorbarxy(x3, y3, STE_fs, STE_fF,{'k' 'k' 'k'})
        hold on
        errorbarxy(x4, y4, STE_ms, STE_mF,{'k' 'k' 'k'})
        hold on
        errorbarxy(x2, y2, STE_ts, STE_tF,{'k' 'k' 'k'})
        hold on
        errorbarxy(x1, y1, STE_us, STE_uF,{'k' 'k' 'k'})
        xlabel('SNR','FontSize',14)
        ylabel('FvFm (initial) ','FontSize',14)
        legend('Control', 'Treatment','FSW','MQ','Location','East')
%         set(gca,'XLim',[0 1.2])
        set(gca,'YLim',[-0.02 0.65])
        hold off
        if option == 1
            set(gcf, 'PaperPosition',[0.25 2.5 8 6])
            set(gca, 'FontSize',12)
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir 'meta' ext
'_SNRFvFm2.png']);
            set(gcf, 'PaperPosition',[0.25 2.5 4 3])
            set(gca, 'FontSize',8)
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir 'meta' ext
'_SNRFvFm2_th.png']);
        end
        %Overlay mean onto scatterplot
        figure
        plot(indx_u,FvFm_u,'.g')
        hold on
        plot(indx_t,FvFm_t,'.r')
        plot(indx_fw,FvFm_fw,'.k')
        plot(indx_mq,FvFm_mq,'.m')
plot(iu_mean,Fu_mean,'g','Marker','diamond','MarkerSize',10,'MarkerFaceColor'
,'g','MarkerEdgeColor','k')
plot(it_mean,Ft_mean,'r','Marker','diamond','MarkerSize',10,'MarkerFaceColor'
,'r','MarkerEdgeColor','k')
plot(iw_mean,Fw_mean,'k','Marker','diamond','MarkerSize',10,'MarkerFaceColor'
,'k','MarkerEdgeColor','k')
plot(im_mean,Fm_mean,'m','Marker','diamond','MarkerSize',10,'MarkerFaceColor'
,'m','MarkerEdgeColor','k')
        hold on
        errorbarxy(x3, y3, STE_fs, STE_fF,{'k' 'k' 'k'})
        hold on
        errorbarxy(x4, y4, STE_ms, STE_mF,{'k' 'k' 'k'})
        hold on
        errorbarxy(x2, y2, STE_ts, STE_tF,{'k' 'k' 'k'})
        hold on
        errorbarxy(x1, y1, STE_us, STE_uF,{'k' 'k' 'k'})
        xlabel('SNR','FontSize',14)
```

```
        ylabel('FvFm initial ','FontSize',14)
        legend('Control', 'Treatment','FSW','MQ','Mean Control','Mean
Treatment','Mean FSW','Mean MQ','Location','East')
        set(gca,'YLim',[-0.02 0.8])
        hold off
        if option == 1
            set(gcf, 'PaperPosition',[0.25 2.5 8 6])
            set(gca, 'FontSize',12)
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir 'meta' ext
'_SNRFvFm3.png']);
            set(gcf, 'PaperPosition',[0.25 2.5 4 3])
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir 'meta' ext
'_SNRFvFm3_th.png']);
        end
        pause
        close all
        %--------Do scatter plot FvFm by calculation -------%
        FvFm_p = [metapp(:,5)];
        FvFm_a = [metapp(:,6)];
        FvFm_w = [metapp(:,7)]; %FvFm WALZ
        FvFm_p1 = (FvFm_p(find(metas(:,13) == 1)));
        FvFm_p2 = (FvFm_p(find(metas(:,13) == 2)));
        FvFm_p3 = (FvFm_p(find(metas(:,13) == 3)));
        FvFm_a1 = (FvFm_a(find(metas(:,13) == 1)));
        FvFm_a2 = (FvFm_a(find(metas(:,13) == 2)));
        FvFm_a3 = (FvFm_a(find(metas(:,13) == 3)));
        FvFm_w1 = (FvFm_w(find(metas(:,13) == 1)));
        FvFm_w2 = (FvFm_w(find(metas(:,13) == 2)));
        FvFm_w3 = (FvFm_w(find(metas(:,13) == 3)));
        x_1indx = repmat(1,1,length(FvFm_p1));
        x_2indx = repmat(2,1,length(FvFm_p2));
        x_3indx = repmat(3,1,length(FvFm_p3));
        x_4indx = repmat(1,1,length(FvFm_a1));
        x_5indx = repmat(2,1,length(FvFm_a2));
        x_6indx = repmat(3,1,length(FvFm_a3));
        x_7indx = repmat(1,1,length(FvFm_w1));
        x_8indx = repmat(2,1,length(FvFm_w2));
        x_9indx = repmat(3,1,length(FvFm_w3));
        x_1l = length(FvFm_p1);
        x_2l = length(FvFm_p2);
        x_3l = length(FvFm_p3);
        x_4l = length(FvFm_a1);
        x_5l = length(FvFm_a2);
        x_6l = length(FvFm_a3);
        x_7l = length(FvFm_w1);
        x_8l = length(FvFm_w2);
        x_9l = length(FvFm_w3);
        figure
        categories = {'Control', 'Treatment',};
        plot(x_1indx,FvFm_p1,'.c')
        hold on
        plot(x_4indx,FvFm_a1,'*b')
        plot(x_7indx,FvFm_w1,'ok')
        plot(x_3indx,FvFm_p3,'.c')
        plot(x_6indx,FvFm_a3,'*b')
        plot(x_9indx,FvFm_w3,'ok')
        ylabel('Dark Adapted Fv/Fm  ','FontSize',14)
```

```
        legend('1 point','Average','WALZ')
        set(gca,'XLim',[0 3])
        set(gca,'YLim',[0 0.8])
        set(gca,'XTick',1:4,'XTickLabel',categories)
        hold off
        if option == 1
            set(gcf, 'PaperPosition',[0.25 2.5 8 6])
            set(gca, 'FontSize',12)
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir 'meta' ext
'_FvFmcompl.png']);
            set(gcf, 'PaperPosition',[0.25 2.5 4 3])
            set(gca, 'FontSize',8)
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir 'meta' ext
'_FvFmcompl_th.png']);
        end
        1*1
        [h,p,ci,stats] = ttest2(FvFm_p1,FvFm_a1)
        1*2
        [h,p,ci,stats] = ttest2(FvFm_p1,FvFm_w1)
        1*3
        [h,p,ci,stats] = ttest2(FvFm_a1,FvFm_w1)
        1*4
        [h,p,ci,stats] = ttest2(FvFm_p2,FvFm_a2)
        1*5
        [h,p,ci,stats] = ttest2(FvFm_p2,FvFm_w2)
        1*6
        [h,p,ci,stats] = ttest2(FvFm_a2,FvFm_w2)
        1*7
        [h,p,ci,stats] = ttest2(FvFm_p3,FvFm_a3)
        1*8
        [h,p,ci,stats] = ttest2(FvFm_p3,FvFm_w3)
        1*9
        [h,p,ci,stats] = ttest2(FvFm_a3,FvFm_w3)
        STE_p1 = nanstd(FvFm_p1)%./sqrt(x_11); %STE for control score (x
axis)
        STE_p2 = nanstd(FvFm_p2)%./sqrt(x_21); %STE for control FvFm (y axis)
        STE_p3 = nanstd(FvFm_p3)%./sqrt(x_31); %STE for treatment score (x
axis)
        STE_a1 = nanstd(FvFm_a1)%./sqrt(x_41); %STE for treatment FvFm (y
axis)
        STE_a2 = nanstd(FvFm_a2)%./sqrt(x_51);
        STE_a3 = nanstd(FvFm_a3)%./sqrt(x_61);
        STE_w1 = nanstd(FvFm_w1)%./sqrt(x_71);
        STE_w2 = nanstd(FvFm_w2)%./sqrt(x_81);
        STE_w3 = nanstd(FvFm_w3)%./sqrt(x_91);
        y_1 = nanmean(FvFm_p1);
        y_2 = nanmean(FvFm_a1);
        y_3 = nanmean(FvFm_w1);
        y_4 = nanmean(FvFm_p2);
        y_5 = nanmean(FvFm_a2);
        y_6 = nanmean(FvFm_w2);
        y_7 = nanmean(FvFm_p3);
        y_8 = nanmean(FvFm_a3);
        y_9 = nanmean(FvFm_w3);
        x_1 = 1;
        x_2 = 2;
        x_3 = 3;
```

```
        if option == 1
            for c = 1:length(total)
                chart = total(c);
                HTMLfile = [HTMLpamdir name ext num2str(c) '.html'];
                clear htmlS
                R = 1;
                htmlS{R} = {['<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">']};R = R +1;
                htmlS{R} = {['<html
xmlns="http://www.w3.org/1999/xhtml">']};R = R +1;
                htmlS{R} = {['<head>']};R = R +1;
                htmlS{R} = {['<meta http-equiv="Content-Type"
content="text/html; charset=utf-8" />']};R = R +1;
                htmlS{R} = {['<title>' matname '</title>']};R = R +1;
                htmlS{R} = {['<style type="text/css">']};R = R +1;
                htmlS{R} = {['<!--']};R = R +1;
                htmlS{R} = {['.style1 {']};R = R +1;
                htmlS{R} = {['font-size: larger;']};R = R +1;
                htmlS{R} = {['font-weight: bold;']};R = R +1;
                htmlS{R} = {['}']};R = R +1;
                htmlS{R} = {['-->']};R = R +1;
                htmlS{R} = {['</style>']};R = R +1;
                htmlS{R} = {['</head>']};R = R +1;
                htmlS{R} = {['<body><p>REV date: ' datestr(now) '</p>']};R =
R +1;
                htmlS{R} = {['']};R = R +1;
                htmlS{R} = {['<center><p class="style1">' matname
'</p></center>']};R = R +1;

                htmlS{R} = {['<p>PWD: ' pwd '</p>']};R = R +1;
                htmlS{R} = {['<p>PWD: ' Fname '</p>']};R = R +1;
                htmlS{R} = {['<p>PWD: ' mfilename '</p>']};R = R +1;
                htmlS{R} = {['<p>']};R = R +1;
                htmlS{R} = {['PAM light curve data plots ']};R = R +1;
                htmlS{R} = {[' ']};R = R +1;
                htmlS{R} = {[' ']};R = R +1;
                htmlS{R} = {[' ']};R = R +1;
                htmlS{R} = {['</p>']};R = R +1;

                %--- Normalized/Smoothed Subplot Overall Comparison ----%
                D = dir([HTMLpamdir name ext num2str(c) '.png']);
                if ~isempty(D)
                    for k = 1:size(D,1)
                        htmlS{R} = {['<td <center><img src = "'
deblank(lower(D(k).name)) '" ></center></td>']};R = R +1;
                    end
                end
                %--- Overall Stat Comparison Table ---------------%
                httmlS{R} = {['<hr/>']};R = R +1;
                htmlS{R} = {['<table border="1" width="95%"> ']};R = R +1;
                htmlS{R} = {['<tr>']};R = R +1;
                htmlS{R} = {['<th align="center">Stats</th>']};R = R +1;
                htmlS{R} = {['<th align="center">Var. Noise_c</th>']};R = R
+1;
                htmlS{R} = {['<th align="center">Var. Noise_t</th>']};R = R
+1;
```

```
                htmlS{R} = {['<th align="center">Var. Peak Tops_c</th>']};R =
R +1;
                htmlS{R} = {['<th align="center">Var. Peak Tops_t</th>']};R =
R +1;
                %                htmlS{R} = {['<th align="center">% Diff.
Var.</th>']};R = R +1;
                htmlS{R} = {['<th align="center">STDEV Noise_c</th>']};R = R
+1;
                htmlS{R} = {['<th align="center">STDEV Noise_t</th>']};R = R
+1;
                htmlS{R} = {['<th align="center">STDEV Peak Tops_c</th>']};R
= R +1;
                htmlS{R} = {['<th align="center">STDEV Peak Tops_t</th>']};R
= R +1;
                %                htmlS{R} = {['<th align="center">% Diff.
STDEV.</th>']};R = R +1;
                htmlS{R} = {['<th align="center">Mean RFU_c</th>']};R = R +1;
                htmlS{R} = {['<th align="center">Mean RFU_t</th>']};R = R +1;
                htmlS{R} = {['<th align="center">Quality_c<_a/th>']};R = R
+1;
                htmlS{R} = {['<th align="center">Quality_t<_a/th>']};R = R
+1;
                htmlS{R} = {['<tr>']};R = R +1;
                htmlS{R} = {['<td align="center">' ' ' '</td>']};R = R +1;
                htmlS{R} = {['<td align="center">' num2str(stats_u(c,5))
'</td>']};R = R +1;
                htmlS{R} = {['<td align="center">' num2str(stats(c,5))
'</td>']};R = R +1;
                htmlS{R} = {['<td align="center">' num2str(stats_u(c,6))
'</td>']};R = R +1;
                htmlS{R} = {['<td align="center">' num2str(stats(c,6))
'</td>']};R = R +1;
                htmlS{R} = {['<td align="center">' num2str(stats_u(c,2))
'</td>']};R = R +1;
                htmlS{R} = {['<td align="center">' num2str(stats(c,2))
'</td>']};R = R +1;
                htmlS{R} = {['<td align="center">' num2str(stats_u(c,3))
'</td>']};R = R +1;
                htmlS{R} = {['<td align="center">' num2str(stats(c,3))
'</td>']};R = R +1;
                htmlS{R} = {['<td align="center">' num2str(stats_u(c,8))
'</td>']};R = R +1;
                htmlS{R} = {['<td align="center">' num2str(stats(c,8))
'</td>']};R = R +1;
                htmlS{R} = {['<td align="center">' num2str(stats_u(c,9))
'</td>']};R = R +1;
                htmlS{R} = {['<td align="center">'
num2str(stats(c,9),'%8.4f') '</td>']};R = R +1;
                htmlS{R} = {['</table>']};R = R +1;
                htmlS{R} = {['<hr/']};R = R +1;
                htmlS{R} = {['</hr>']};R = R +1;
                %--- Normalized/Smoothed Subplot Peak Comparison -------%
                D = dir([HTMLpamdir name '_c' num2str(chart) '_p'  '*.png']);
                if ~isempty(D)

                    htmlS{R} = {['<hr/>']};R = R +1;
                    htmlS{R} = {['<table border="0.5" width="95%"> ']};R = R
```

```
+1;
                    htmlS{R} = {['<tr>']};R = R +1;
                    htmlS{R} = {['<td <center><a href="' [name '_c'
num2str(c) '_p' '1.png'] '"><img src = "' [name '_c' num2str(c) '_p'
'1th.png'] '" ></a><center></td>']};R = R +1;
                    htmlS{R} = {['<td <center><a href="' [name '_c'
num2str(c) '_p' '2.png'] '"><img src = "' [name '_c' num2str(c) '_p'
'2th.png'] '" ></a><center></td>']};R = R +1;
                    htmlS{R} = {['<tr>']};R = R +1;
                    htmlS{R} = {['<td <center><a href="' [name '_c'
num2str(c) '_p' '3.png'] '"><img src = "' [name '_c' num2str(c) '_p'
'3th.png'] '" ></a><center></td>']};R = R +1;
                    htmlS{R} = {['<td <center><a href="' [name '_c'
num2str(c) '_p' '4.png'] '"><img src = "' [name '_c' num2str(c) '_p'
'4th.png'] '" ></a><center></td>']};R = R +1;
                    htmlS{R} = {['<tr>']};R = R +1;
                    htmlS{R} = {['<td <center><a href="' [name '_c'
num2str(c) '_p' '5.png'] '"><img src = "' [name '_c' num2str(c) '_p'
'5th.png'] '" ></a><center></td>']};R = R +1;
                    htmlS{R} = {['<td <center><a href="' [name '_c'
num2str(c) '_p' '6.png'] '"><img src = "' [name '_c' num2str(c) '_p'
'6th.png'] '" ></a><center></td>']};R = R +1;
                    htmlS{R} = {['<tr>']};R = R +1;
                    htmlS{R} = {['<td <center><a href="' [name '_c'
num2str(c) '_p' '7.png'] '"><img src = "' [name '_c' num2str(c) '_p'
'7th.png'] '" ></a><center></td>']};R = R +1;
                    htmlS{R} = {['<td <center><a href="' [name '_c'
num2str(c) '_p' '8.png'] '"><img src = "' [name '_c' num2str(c) '_p'
'8th.png'] '" ></a><center></td>']};R = R +1;
                    htmlS{R} = {['<tr>']};R = R +1;
                    htmlS{R} = {['<td <center><a href="' [name '_c'
num2str(c) '_p' '9.png'] '"><img src = "' [name '_c' num2str(c) '_p'
'9th.png'] '" ></a><center></td>']};R = R +1;
                    htmlS{R} = {['<td <center><a href="' [name '_c'
num2str(c) '_p' '10.png'] '"><img src = "' [name '_c' num2str(c) '_p'
'10th.png'] '" ></a><center></td>']};R = R +1;
                    htmlS{R} = {['<tr>']};R = R +1;
                    htmlS{R} = {['<td <center><a href="' [name '_c'
num2str(c) '_p' '11.png'] '"><img src = "' [name '_c' num2str(c) '_p'
'11th.png'] '" ></a><center></td>']};R = R +1;
                    htmlS{R} = {['<td <center><a href="' [name '_c'
num2str(c) '_p' '12.png'] '"><img src = "' [name '_c' num2str(c) '_p'
'12th.png'] '" ></a><center></td>']};R = R +1;
                    htmlS{R} = {['<tr>']};R = R +1;
                    htmlS{R} = {['<td <center><a href="' [name '_c'
num2str(c) '_p' '13.png'] '"><img src = "' [name '_c' num2str(c) '_p'
'13th.png'] '" ></a><center></td>']};R = R +1;
                    htmlS{R} = {['<td <center> ' ' <center></td>']};R = R +1;
                    %           htmlS{R} = {['<td align="center">'
num2str(fval(B,10),'%8.4f') '</td>']};R = R +1;
                    htmlS{R} = {['<tr>']};R = R +1;
                    htmlS{R} = {['</table>']};R = R +1;
                end
                %-- Peak Comparison Calculations ------%
                htmlS{R} = {['<hr/']};R = R +1;
                htmlS{R} = {['</hr>']};R = R +1;
                httmlS{R} = {['<hr/>']};R = R +1;
```

```
                htmlS{R} = {['<table border="1" width="95%"> ']};R = R +1;
                htmlS{R} = {['<tr>']};R = R +1;
                htmlS{R} = {['<th align="center">Peak number</th>']};R = R
+1;
                htmlS{R} = {['<th align="center">FvFm_c</th>']};R = R +1;
                htmlS{R} = {['<th align="center">FvFm_t</th>']};R = R +1;
                htmlS{R} = {['<th align="center">% Diff FvFm c:t</th>']};R =
R +1;
                htmlS{R} = {['<th align="center">std_c</th>']};R = R +1;
                htmlS{R} = {['<th align="center">std_t</th>']};R = R +1;
                htmlS{R} = {['<th align="center">Diff std</th>']};R = R +1;
                htmlS{R} = {['<th align="center">var_c</th>']};R = R +1;
                htmlS{R} = {['<th align="center">var_t</th>']};R = R +1;
                htmlS{R} = {['<th align="center">% Diff var c:t</th>']};R = R
+1;
                htmlS{R} = {['<th align="center">% peak height_c</th>']};R =
R +1;
                htmlS{R} = {['<th align="center">% peak height_t</th>']};R =
R +1;
                htmlS{R} = {['<tr>']};R = R +1;
                for B = 1:length(PBINS1)
                    htmlS{R} = {['<tr>']};R = R +1; %may or may not need
                    if B >=2 & B <= 9
                        htmlS{R} = {['<td align="center" bgcolor="#FF0000">'
num2str(p_comp(PBINS1(B),2)) '</td>']};R = R +1; %num2str(fval(B,2))
                    else
                        htmlS{R} = {['<td align="center"
bgcolor="#0000FF"><font color="#FFFFFF">' num2str(p_comp(PBINS1(B),2))
'</td>']};R = R +1;
                    end
                    htmlS{R} = {['<td align="center">'
num2str(p_comp(PBINS1(B),3)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(p_comp(PBINS1(B),4)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(p_comp(PBINS1(B),5)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(p_comp(PBINS1(B),6)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(p_comp(PBINS1(B),7)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(p_comp(PBINS1(B),8)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(p_comp(PBINS1(B),9)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(p_comp(PBINS1(B),10)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(p_comp(PBINS1(B),11)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(p_comp(PBINS1(B),12)) '</td>']};R = R +1;
                    htmlS{R} = {['<td align="center">'
num2str(p_comp(PBINS1(B),13),'%8.4f') '</td>']};R = R +1;
                end
                htmlS{R} = {['</table>']};R = R +1;
                PBINS1 = (13*c)+1:(13*(c+1));
                htmlS{R} = {['<hr/']};R = R +1;
                htmlS{R} = {['</hr>']};R = R +1;
```

```
                %----- End of tables and graphs ------%
                htmlS{R} = {['<br/>']};R = R +1;
                htmlS{R} = {['<p>REV date: ' datestr(now) '</p>']};R = R +1;
                htmlS{R} = {['</body>']};R = R +1;
                htmlS{R} = {['</html>']};R = R +1;
                fid = fopen(HTMLfile,'wt');
                if fid > 0
                    [M N] = size(htmlS);
                    for S = 1:N
                        fwrite(fid,[char(htmlS{S}) 10],'char');
                    end
                    fclose(fid);;
                else
                    disp([FCNname ' Error (Cound not open' HTMLfile ')'])
                end
                disp(['  Created ' HTMLfile])
            end
        end
    end
else
    disp('Not a valid code')
end
end
```

metastats_bfvals_.m

```
function [x,y] = metastats_bfvals(step,normal,option);
%metastats_bfvals.m
%For importing the baseline and all peak data for other stats.
% Created: 21 Nov 2013 by N. Bobco
sysvars_;
if nargin == 1
    option = 0;
end
if ~exist(FPpamdir)
    mkdir(FPpamdir)
end
if step == 1
    if normal == 1
        ext = '_ns'
        wildcard = '*_fp.mat'
        disp([pwd])
        f = dir(wildcard)
        p = 1;
        c = 1;
        z = 1;
        for F = 1:size(f,1)

            name = (f(F).name)
            load([FPpamdir name])
            for r = 1:size(fval);
                metaf(p,:) = [F fval(r,1) fval(r,2) fval(r,3) fval(r,4)
fval(r,5) fval(r,6) fval(r,7) fval(r,8) fval(r,9) fval(r,10) fval(r,11)
fval(r,12) fval(r,13) fval(r,14)];
                p = p+1;
            end
            ititle = name(1:end-10);
```

```
            for l = 1:length(chartindx)
                type = input('Enter: 1 = uptake, 2 = T0, 3 = T, 4 = MQ, 5 =
FSW: ');
                indx(z,:) = [F l type];
                z = z+1;
            end
            index(c,1) =[{ititle}];
            c = c + 1;
        end
        if option == 1
            save(['meta' ext '_msb.mat'],'metaf','index','indx');
        end
    end
else step == 2
    if normal == 1
        ext = '_ns'
        close all
        matname = ['meta_ns_msb.mat'];
        Fname = getfname(matname)
        load([FPpamdir Fname]);
        %---- Baseline slope ---------%
        slopes = [metal(:,2)];
        yint = [metal(:,4)];
        slope_u= (slopes(find(metal(:,3) == 1)));
        yint_u = (yint(find(metal(:,3) == 1)));
        s_u = nanmean(slope_u)
        yint_mean = nanmean(yint_u)
        a = length(slope_u);
        slope_t0= (slopes(find(metal(:,3) == 2)));
        s_t0 = nanmean(slope_t0);
        b = length(slope_t0);
        slope_t= (slopes(find(metal(:,3) == 3)));
        s_t = nanmean(slope_t);
        c = length(slope_t);
        slope_m= (slopes(find(metal(:,3) == 4)));
        s_m = nanmean(slope_m);
        d = length(slope_m);
        slope_f= (slopes(find(metal(:,3) == 5)));
        s_f = nanmean(slope_f);
        e = length(slope_f);
        x_su = repmat(1,1,a);
        x_st0= repmat(2,1,b);
        x_st = repmat(3,1,c);
        x_sf = repmat(4,1,d);
        x_sm = repmat(5,1,e);
        figure
        categories = {'Control','T0','Treatment','FSW','MQ'};
        plot(x_su,slope_u,'.g')
        hold on
        plot(x_st0,slope_t0,'.b')
        plot(x_st,slope_t,'.r')
        plot(x_sf,slope_f,'.k')
        plot(x_sm,slope_m,'.m')
        ylabel('Baseline Slope ','FontSize',14)
        set(gca,'XLim',[0 6])
        set(gca,'XTick',1:5,'XTickLabel',categories)
        hold off
```

```
        if option == 1
            set(gcf, 'PaperPosition',[0.25 2.5 8 6])
            set(gca, 'FontSize',12)
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir 'meta' ext
'_slope1.png']);
            set(gcf, 'PaperPosition',[0.25 2.5 4 3])
            set(gca, 'FontSize',8)
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir 'meta' ext
'_slope1_th.png']);
        end
        1*1
        [h,p,ci,stats] = ttest(slope_u,slope_t)
        1*2
        [h,p,ci,stats] = ttest2(slope_t,slope_m)
        1*3
        [h,p,ci,stats] = ttest2(slope_t,slope_f)
        1*4
        [h,p,ci,stats] = ttest2(slope_m,slope_f)
        1*5
        [h,p,ci,stats] = ttest2(slope_u,slope_m)
        1*6
        [h,p,ci,stats] = ttest2(slope_u,slope_f)

        STE_su = nanstd(slope_u) %./sqrt(a) %STE for control score (x axis)
        STE_st = nanstd(slope_t) %./sqrt(b); %STE for treatment score (x ax)
        STE_sm = nanstd(slope_m) %./sqrt(d); %STE for treatment FvFm (y axis)
        STE_sf = nanstd(slope_f) %./sqrt(e);
        x1 = 1;
        x2 = 2;
        x3 = 3;
        x4 = 4;
        y1 = nanmean(slope_u)
        y2 = nanmean(slope_t)
        y4 = nanmean(slope_m)
        y3 = nanmean(slope_f)
        figure
        categories = {'Control','Treatment','FSW','MQ'};
plot(x1,y1,'g','Marker','diamond','MarkerSize',20,'MarkerFaceColor','g','Mark
erEdgeColor','k')
        hold on
plot(x2,y2,'r','Marker','diamond','MarkerSize',20,'MarkerFaceColor','r','Mark
erEdgeColor','k')
plot(x3,y3,'k','Marker','diamond','MarkerSize',20,'MarkerFaceColor','k','Mark
erEdgeColor','k')
plot(x4,y4,'m','Marker','diamond','MarkerSize',20,'MarkerFaceColor','m','Mark
erEdgeColor','k')
        hold on
        errorbar(x1, y1, STE_su,'k')
        errorbar(x2, y2, STE_st,'k')
        errorbar(x3, y3, STE_sf,'k')
        errorbar(x4, y4, STE_sm,'k')
        ylabel('Mean Baseline Slope ','FontSize',14)
        set(gca,'XLim',[0 5])
        set(gca,'XTick',1:4,'XTickLabel',categories)
        hold off
        if option == 1
            set(gcf, 'PaperPosition',[0.25 2.5 8 6])
```

```
            set(gca, 'FontSize',12)
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir 'meta' ext
'_slope2.png']);
            set(gcf, 'PaperPosition',[0.25 2.5 4 3])
            set(gca, 'FontSize',8)
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir 'meta' ext
'_slope2_th.png']);
        end
        %----- FvFm 1st & last -------%
        FvFms = [metaf(:,13) metaf(:,14) metaf(:,15)];
        for z = 1:length(FvFms)
            if FvFms(z,2) < 0
                FvFms(z,2) = 0;
            else
                FvFms(z,2) = FvFms(z,2);
            end
        end
        %----- FvFm by peak (C,T0,T) -------%
        FvFm_p = [FvFms(:,1)];
        FvFm_a = [FvFms(:,2)];
        FvFm_w = [FvFms(:,3)];
        xaxis = [metaf(:,3)];
        FvFm_up = (FvFm_p(find(metaf(:,16) == 1)));
        FvFm_ua = (FvFm_a(find(metaf(:,16) == 1)));
        FvFm_uw = (FvFm_w(find(metaf(:,16) == 1)));
        x_u = (xaxis(find(metaf(:,16) == 1)));
        FvFm_t0p = (FvFm_p(find(metaf(:,16) == 2)));
        FvFm_t0a = (FvFm_a(find(metaf(:,16) == 2)));
        FvFm_t0w = (FvFm_w(find(metaf(:,16) == 2)));
        x_t0 = (xaxis(find(metaf(:,16) == 2)));
        FvFm_tp = (FvFm_p(find(metaf(:,16) == 3)));
        FvFm_ta = (FvFm_a(find(metaf(:,16) == 3)));
        FvFm_tw = (FvFm_w(find(metaf(:,16) == 3)));
        x_t = (xaxis(find(metaf(:,16) == 3)));
        FvFm_mp = (FvFm_p(find(metaf(:,16) == 4)));
        FvFm_ma = (FvFm_a(find(metaf(:,16) == 4)));
        FvFm_mw = (FvFm_w(find(metaf(:,16) == 4)));
        x_m = (xaxis(find(metaf(:,16) == 4)));
        FvFm_fp = (FvFm_p(find(metaf(:,16) == 5)));
        FvFm_fa = (FvFm_a(find(metaf(:,16) == 5)));
        FvFm_fw = (FvFm_w(find(metaf(:,16) == 5)));
        x_f = (xaxis(find(metaf(:,16) == 5)));
        xu = x_u';
        xt0 = x_t0';
        xt = x_t';
        xm = x_m';
        xf = x_f';
        figure
        plot(xu,FvFm_up,'.g')
        hold on
        plot(xu,FvFm_ua,'.b')
        plot(xu,FvFm_uw,'.k')
        xlabel('Peak','FontSize',14)
        ylabel('FvFm','FontSize',14)
        hold off
        if option == 1
            set(gcf, 'PaperPosition',[0.25 2.5 8 6])
```

```
            set(gca, 'FontSize',12)
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir 'meta' ext
'_allFvFms1.png']);
            set(gcf, 'PaperPosition',[0.25 2.5 4 3])
            set(gca, 'FontSize',8)
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir 'meta' ext
'_allFvFms1_th.png']);
        end
    end
end
end
```

smooth_everything.m

```
function [datacol]=smooth_everything(name,step,option);
%smooth_everything.m
%Combo of smmoth_noise_fnc only smooth over peaks, then goes into peak
%finding for find_peaks_fnc and tries to calculate FvFms for those data.
%Used for metastats_smooth.m comparison program of calc FvFm.
%Created: 21 Nov 2013 by N.Bobco
sysvars_;
if ~exist(HTMLpamdir)
    mkdir(pwd,HTMLpamdir)
end
if nargin == 1
    option = 0;
end
if step == 1
    matname = [name '_pr.mat'];
    Fname = getfname(matname)
    load([FPpamdir Fname]);
    ext = '_smooth';
    for k = 1:length(pamp)
        PM = settings(k,3);
        if PM < 15;
            Fo_w = pamp(k,2)*(1.5^(15-PM));
            Fm_w = pamp(k,3)*(1.5^(15-PM));
            FvFm_w = pamp(k,4);
            fval_w(k,:) = [k Fo_w Fm_w FvFm_w];
        elseif (PM >15) & (PM <20);
            Fo_w = pamp(k,2)/(1.5^(PM-15));
            Fm_w = pamp(k,3)/(1.5^(PM-15));
            FvFm_w = pamp(k,4);
            fval_w(k,:) = [k Fo_w Fm_w FvFm_w];
        elseif (PM >=20) & (PM <25);
            Fo_w = 10*(pamp(k,2)/(1.5^(PM-15)));
            Fm_w = 10*(pamp(k,3)/(1.5^(PM-15)));
            FvFm_w = pamp(k,4);
            fval_w(k,:) = [k Fo_w Fm_w FvFm_w];
        elseif PM >= 25;
            Fo_w = 100*(pamp(k,2)/(1.5^(PM-15)));
            Fm_w = 100*(pamp(k,3)/(1.5^(PM-15)));
            FvFm_w = pamp(k,4);
            fval_w(k,:) = [k Fo_w Fm_w FvFm_w];
        else PM == 15;
            Fo_w = pamp(k,2);
            Fm_w = pamp(k,3);
```

```
            FvFm_w = pamp(k,4);
            fval_w(k,:) = [k Fo_w Fm_w FvFm_w];
        end
    end
    for k = 1:length(fluorescence(1,:))
        chartnum = settings(:,2); %indexing chart number for correct PM gain
        indx = find(chartnum >=k,1);
        PM = settings(indx,3);
        if PM < 15;
            data(:,k) = fluorescence(:,k)*(1.5^(15-PM));
        elseif (PM >15) & (PM <20);
            data(:,k) = (fluorescence(:,k)/(1.5^(PM-15)));
        elseif (PM >=20) & (PM <25);
            data(:,k) = 10*(fluorescence(:,k)/(1.5^(PM-15)));
        elseif PM >= 25;
            data(:,k) = 100*(fluorescence(:,k)/(1.5^(PM-15)));
        else PM == 15;
            data(:,k) = fluorescence(:,k);
        end
    end
    for c = 1:length(fluorescence(1,:))
        datacol = data(:,c);

        figure(c)
        plot(fluorescence(:,c))
        xlabel('Time (min)', 'FontSize',14)
        ylabel('Relative Fluorescence', 'FontSize',14)
        title('Raw Data')
        set(gca,'XLim',[0 13000]) %set max value for 8 min chart length
        set(gca,'XTick',[0:1500:20000]) %about 1600 points per minute.
        set(gca,'XTickLabel',['0';'1';'2';'3';'4';'5';'6';'7';'8'])
        if option == 1
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir name ext '_r'
num2str(c) '.png']) %for smooth test
        end
        figure(c)
        plot(datacol)
        xlabel('Time (min)', 'FontSize',14)
        ylabel('Relative Fluorescence', 'FontSize',14)
        title('Normalized Data')
        set(gca,'XLim',[0 13000]) %set max value for 8 min chart length
        set(gca,'XTick',[0:1500:20000]) %about 1600 points per minute.
        set(gca,'XTickLabel',['0';'1';'2';'3';'4';'5';'6';'7';'8'])
        if option == 1
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir name ext '_20n_'
num2str(gcf) '.png']) %for smooth test
            set(gcf, 'PaperPosition',[0.25 2.5 4 3])
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir name ext '_20n_'
num2str(gcf) 'th' '.png'])
        end
        clear figure
        span=50; %size of the averaging
        window=ones(span,1)/span;
        datasf=convn(datacol(:,1),window,'same'); %smoothing all
        figure(c)
        plot(datasf, 'r')
        xlabel('Time (min)', 'FontSize',14)
```

```
        ylabel('Relative Fluorescence', 'FontSize',14)
        title('Normalized and Smoothed Noise')
        set(gca,'XLim',[0 13000]) %set max value for 8 min chart length
        set(gca,'XTick',[0:1500:20000]) %about 1600 points per minute.
        set(gca,'XTickLabel',['0';'1';'2';'3';'4';'5';'6';'7';'8'])
        if option == 1
            set(gcf, 'PaperPosition',[0.25 2.5 8 6]);
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir name ext '_20ns_'
num2str(gcf) '.png']);
            set(gcf, 'PaperPosition',[0.25 2.5 4 3]);
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir name ext '_20ns_'
num2str(gcf) 'th' '.png']);
        end
        smoothnormdata(:,c) = datasf;
        clear chart
        close all
    end
    if option == 1
        save([name ext '_ns.mat'],'smoothnormdata','pamp','fval_w')
    end
elseif step == 2
    matname = [name '_smooth_ns.mat'];
    Fname = getfname(matname)
    load([FPpamdir Fname]);
    ext = '_smooth';
    M = length(smoothnormdata);
    N = length(smoothnormdata(1,:));
    peakindx = repmat(0,M,N);
    peakindx2 = repmat(0,M,N);
    pp = 1;
    for c = 1:length(smoothnormdata(1,:))
        chart = c;
        data = smoothnormdata(:,c);
        hold off
        figure(c)
        multicolorplot(data,data,100:6150)
        set(gca,'XLim',[0 13000]) %set max value for 8 min chart length
        set(gca,'XTick',[0:1500:30000]) %about 1500 points per minute.
        set(gca,'XTickLabel',['0';'1';'2';'3';'4';'5';'6';'7';'8'])
        if option == 1
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir name ext '_'
num2str(chart) '_c' num2str(c) '.png'])
        end
        FH = figure;  %start of peak plots
        plot(data(:,1), '.b-')
        xlim([0 13000])
        hold on
        PBINS1 = [  1 700 1450 2200 3000 3750 4500 5300 6050 6300 7070 8600
11500];
        PBINS2 = [100 850 1650 2400 3200 3925 4700 5450 6250 6450 7220 8800
11850];
        for B = 1:length(PBINS1);
            er = PBINS1(B):PBINS2(B);
            peak = data(er,1);
            peakindx(er,c) = 1;
            peakindx2(er,c) = B;
            figure(2)
```

```
            [x,i] = nanmax(peak);
            eri = i-50:i;    %changed to i-50 from i-100
            eri = eri(find(eri > 0));
            plot(er,peak,'r-')
            plot(er(eri),peak(eri),'.g-')
            title(['Peak ',num2str(B)],'FontSize',8);
            erl = eri;
            [x,il] = nanmax(abs(diff(peak(erl))));
            err = i+10:i+30;
            err = err(find(err > 0 & err <= length(er))); %had as
if ~isempty(err);
                [x,ir] = nanmax(abs(diff(peak(err))));
                figure(FH);
                eri2 = erl(il)-15:err(ir)+15;
                eri2 = eri2(find(eri2 > 0 & eri2 <=length(er)));
                plot(er(eri2),peak(eri2),'.c-')
                xlim([PBINS1(B)-20 PBINS2(B)+20])
                if ~isempty(eri2);
                    data(er(eri),2) = B;
                    Fo = peak(eri2(1));
                    Fo_a = mean(peak(eri(2):eri2(1)));
                    Fm = data(er(i));
                    if B == 1
                        Fm_a=mean(peak(eri2(10:20)));
                        peakindx(er(eri2(10:20)),c) = 2;
                        hold on
                        plot(er(eri(2):eri2(1)),peak(eri(2):eri2(1)),'.g-')
                        plot(er(eri2(1:20)), peak(eri2(1:20)), '.m-')
                    else
                        if length(eri2) > 45
                            Fm_a = mean(peak(eri2(30:45)));
                            peakindx(er(eri2(30:45)),c) = 2;
                            if eri(2) > PBINS1(B)
                                hold on
                                plot(er(eri(2)-10:eri(2)-3),peak(eri(2)-
10:eri(2)-3),'.g-')

                                plot(er(eri2(30:45)), peak(eri2(30:45)), '.m-
')

                            else
                                plot(er(eri2(30:45)), peak(eri2(30:45)), '.m-
')

                            end
                        elseif length(eri2) > 10 & length(eri2) <45
                            Fm_a=mean(peak(eri2(20:25)));
                            peakindx(er(eri2(20:25)),c) = 2;
                            if eri(2) > PBINS1(B)
                                hold on
                                plot(er(eri(2)-5:eri2(1)),peak(eri(2)-
5:eri2(1)),'.g-')

                                plot(er(eri2(20:25)), peak(eri2(20:25)), '.m-
')

                            else
                                plot(er(eri2(20:25)), peak(eri2(20:25)), '.m-
')

                            end
                        else
                            Fm_a=mean(peak(eri2(5:10)));
```

```
                        peakindx(er(eri2(5:10)),c) = 2;
                        if eri(2) > PBINS1(B)
                            hold on
   plot(er(eri(2):eri2(1)),peak(eri(2):eri2(1)),'.g-')
                            plot(er(eri2(5:10)), peak(eri2(5:10)), '.m-')
                        else
                            plot(er(eri2(5:10)), peak(eri2(5:10)), '.m-')
                        end
                    end
                end

                Fv = Fm-Fo;
                Fv_a = (Fm_a-Fo_a);
                FvFm = Fv./Fm;
                FvFm_a = Fv_a./Fm_a;
                PAR = pamp(pp,10);
                ETR = PAR.*FvFm.*0.85*0.5;
                ETR_a = PAR.*FvFm_a.*0.85*0.5;
                data(eri,2) = B;
                if option == 1
                    set(gcf, 'PaperPosition',[0.25 2.5 8 6])
                    set(gca, 'FontSize',10)
                    print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir name
ext '_' num2str(chart) '_c' num2str(c) '_p' num2str(B) '.png']);
                    set(gcf, 'PaperPosition',[0.25 2.5 2 1.5])
                    set(gca, 'FontSize',8)
                    print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir name
ext '_' num2str(chart) '_c' num2str(c) '_p' num2str(B) 'th.png']);
                end
            end
        else
            Fo = NaN;
            Fo_a = NaN;
            Fm = NaN;
            Fm_a = NaN;
            Fv = NaN;
            Fv_a = NaN;
            FvFm = NaN;
            FvFm_a = NaN;
            ETR = NaN;
            ETR_a = NaN;
        end
        Fo_w = fval_w(pp,2); %WALZ Fo
        Fm_w = fval_w(pp,3); %WALZ Fm
        Fv_w = Fm_w-Fo_w;
        FvFm_w = pamp(pp,4); %WALZ FvFm
        qP_w = (pamp(pp,6))./1000; %WALZ qP
        NPQ_w = (pamp(pp,5))./1000;%WALZ NPQ
        PAR = pamp(pp,10);
        ETR_w = pamp(pp,5)./10; %WALZ ETR
        fval(pp,:) = [c B Fo Fo_a Fo_w Fm Fm_a Fm_w Fv Fv_a Fv_w FvFm
FvFm_a FvFm_w];
        lc(pp,:) = [c B qP_w NPQ_w PAR ETR ETR_a ETR_w];
        Fo = NaN;
        Fo_a = NaN;
        Fm = NaN;
        Fm_a = NaN;
```

```
            Fv = Fm-Fo;
            Fv_a = NaN;
            FvFm = NaN;
            FvFm_a = NaN;
            ETR = NaN;
            ETR_a = NaN;
            peak(1:eri(end)) = NaN;
            close all
            figure(FH)
            plot(data(:,1), '.b-')
            xlim([0 13000])
            hold on
            pp = pp+1;
        end
        pp = (13*c)+1;
        close all
    end
    if option == 1
        save([name ext
'_fp.mat'],'smoothnormdata','fval','pamp','fval_w','lc','peakindx','peakindx2
') %sn for normalized and smoothed
    end
end
end
```

metastats_smooth_.m

```
function metastats_smooth(name,step,type,number,option);
%metastats_smooth_.m
%Like metastats, only with using smoothed all over data,
close all
sysvars_;
if ~exist(HTMLpamdir)
    mkdir(pwd,HTMLpamdir)
end
if nargin == 1
    option = 0;
end
if step == 1 %import smoothed or gaussian or treatment
    if type == 1 %for smoothed all over data
        matname = [name '_fp.mat'];
        Fname = getfname(matname)
        load([FPpamdir Fname])
        ext = '_smooth';
        eval([ 'smoothnormdata' '_u' ' = smoothnormdata;' ]);
        eval([ 'peakindx' '_u' ' = peakindx;' ]);
        eval([ 'peakindx2' '_u' ' = peakindx2;' ]);
        eval([ 'fval' '_u' ' = fval;' ]);
        eval([ 'lc' '_u' ' = lc;' ]);
        matname = [name '_smooth_fp.mat'];
        Fname = getfname(matname)
        load([FPpamdir Fname]);
        total = [1:1:number];
        for c = 1:length(total)
            choice = input('For this uptake chart number, write numerical
treatment chart number for comparison: ');
            u_data = smoothnormdata_u(:,c);
```

```
            t_data = smoothnormdata(:,choice);
            u = (1:length(u_data))';
            t = (1:length(t_data))';
            figure(c)
            subplot(2,1,1)
            multicolorplot(u_data,u_data,100:6150)
            ylabel('Relative Fluorescence')
            title('No Peak Smoothing', 'FontSize',12)
            set(gca,'XLim',[0 13000]) %set max value for 8 min chart length
            set(gca,'XTick',[0:1500:30000]) %about 1500 points per minute.
            set(gca,'XTickLabel',['0';'1';'2';'3';'4';'5';'6';'7';'8'])
            subplot(2,1,2)
            multicolorplot(t_data,t_data,100:6150)
            xlabel('Time (min)')
            ylabel('Relative Fluorescence')
            title('Peaks Smoothed (boxcar average)')
            set(gca,'XLim',[0 13000]) %set max value for 8 min chart length
            set(gca,'XTick',[0:1500:30000]) %about 1500 points per minute.
            set(gca,'XTickLabel',['0';'1';'2';'3';'4';'5';'6';'7';'8'])
            if option == 1
                set(gcf, 'PaperPosition',[0.25 2.5 8 6])
                print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir name ext '_ns_'
num2str(c) '.png']);
            end
            clear figure
        end
    elseif type == 2 %for gaussian data
    elseif type == 3 %treatment
        matname = [name '_fp.mat'];
        Fname = getfname(matname)
        load([FPpamdir Fname])
        ext = '_concent2';
        eval([ 'smoothnormdata' '_u' ' = smoothnormdata;' ]);
        matname = ['DRD100x3_fp.mat'];
        Fname = getfname(matname)
        load([FPpamdir Fname]);
        total = [1:1:number];
        for c = 1:length(total)
            choice = input('For this uptake chart number, write numerical
treatment chart number for comparison: ');
            u_data = smoothnormdata_u(:,c);
            t_data = smoothnormdata(:,choice);
            u = (1:length(u_data))';
            t = (1:length(t_data))';
            figure(c)
            subplot(2,1,1)
            multicolorplot(u_data,u_data,100:6150)
            ylabel('Relative Fluorescence')
            title('Unconcentrated', 'FontSize',12)
            set(gca,'XLim',[0 13000]) %set max value for 8 min chart length
            set(gca,'XTick',[0:1500:30000]) %about 1500 points per minute.
            set(gca,'XTickLabel',['0';'1';'2';'3';'4';'5';'6';'7';'8'])
            subplot(2,1,2)
            multicolorplot(t_data,t_data,100:6150)
            xlabel('Time (min)')
            ylabel('Relative Fluorescence')
            title('Concentrated')
```

```
            set(gca,'XLim',[0 13000]) %set max value for 8 min chart length
            set(gca,'XTick',[0:1500:30000]) %about 1500 points per minute.
            set(gca,'XTickLabel',['0';'1';'2';'3';'4';'5';'6';'7';'8'])
            if option == 1
                set(gcf, 'PaperPosition',[0.25 2.5 8 6])
                print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir name ext '_ns_'
num2str(c) '.png']);
            end
            clear figure
        end
    else
        disp('not a valid code')
    end
elseif step == 2 %plotting data, get from meta stats orig?

    if type == 1
        ext = '_smooth'
        wildcard = '*_smooth_fp.mat'
        disp([pwd])
        f = dir(wildcard)
        p = 1;
        c = 1;
        j = 1;
        for F = 1:size(f,1)
            name = (f(F).name)
            load([FPpamdir name])
            for r = 1:size(fval);
                type = input('Enter: 1 = uptake, 2 = T0, 3 = T, 4 = MQ, 5 =
FSW: ');
                metaf_s(p,:) = [F fval(r,1) fval(r,2) fval(r,3) fval(r,4)
fval(r,5) fval(r,6) fval(r,7) fval(r,8) fval(r,9) fval(r,10) fval(r,11)
fval(r,12) fval(r,13) fval(r,14) type];
                p = p+1;
            end
            titles = name(1:end-10);
            index(c,1) =[{titles}];
            c = c + 1;
        end
        if option == 1
            save(['meta' ext '_ms.mat'],'metaf_s','index');
        end
    elseif type == 2

    end
elseif step == 3
    if type == 1
        matname = [name '_smooth_ms.mat'];
        Fname = getfname(matname)
        load([FPpamdir Fname])
        ext = '_smooth';
        FvFms_o = [metaf_o(:,13) metaf_o(:,14) metaf_o(:,15)];
        for z = 1:length(FvFms_o)
            if FvFms_o(z,2) < 0
                FvFms_o(z,2) = 0;
            else
                FvFms_o(z,2) = FvFms_o(z,2);
            end
```

```
        end
        FvFms_s = [metaf_s(:,13) metaf_s(:,14) metaf_s(:,15)];
        for z = 1:length(FvFms_s)

            if FvFms_s(z,2) < 0
                FvFms_s(z,2) = 0;
            else
                FvFms_s(z,2) = FvFms_s(z,2);
            end
        end
        FvFm_ao = [FvFms_o(:,1)];
        FvFm_as = [FvFms_s(:,1)];
        xaxis = [metaf_o(:,3)];
        FvFm_uao = (FvFm_ao(find(metaf_o(:,3) == 2 & metaf_o(:,16) == 1)));
        FvFm_uas = (FvFm_as(find(metaf_s(:,3) == 2 & metaf_s(:,16) == 1)));
        a = length(FvFm_uao);
        FvFm_t0ao = (FvFm_ao(find(metaf_o(:,3) == 2 & metaf_o(:,16) == 2)));
        FvFm_t0as = (FvFm_as(find(metaf_s(:,3) == 2 & metaf_s(:,16) == 2)));
        b = length(FvFm_t0ao);
        FvFm_tao = (FvFm_ao(find(metaf_o(:,3) == 2 & metaf_o(:,16) == 3)));
        FvFm_tas = (FvFm_as(find(metaf_s(:,3) == 2 & metaf_s(:,16) == 3)));
        c = length(FvFm_tao);
        x_su = repmat(1,1,a);
        x_st0= repmat(2,1,b);
        x_st = repmat(3,1,c);
        figure
        categories = {'Control','T0','Treatment'};
        plot(x_su,FvFm_uao,'.c')
        hold on
        plot(x_su,FvFm_uas,'.r')
        plot(x_st0,FvFm_t0ao,'.c')
        plot(x_st0,FvFm_t0as,'.r')
        plot(x_st,FvFm_tao,'.c')
        plot(x_st,FvFm_tas,'.r')
        ylabel('Fv/Fm (peak2) ','FontSize',14)
        legend('Raw Peaks','Peaks Smoothed (boxcar)')
        set(gca,'XLim',[0 4])
        set(gca,'XTick',1:3,'XTickLabel',categories)
        hold off
        if option == 1
            set(gcf, 'PaperPosition',[0.25 2.5 8 6])
            set(gca, 'FontSize',12)
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir 'meta' ext
'_FvFm1.png']);
            set(gcf, 'PaperPosition',[0.25 2.5 4 3])
            set(gca, 'FontSize',8)
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir 'meta' ext
'_FvFm1_th.png']);
        end
        1*1
        [h,p,ci,stats] = ttest(FvFm_uao,FvFm_uas)
        1*2
        [h,p,ci,stats] = ttest(FvFm_tao,FvFm_tas)
        1*3
        [h,p,ci,stats] = ttest(FvFm_uao,FvFm_tao)
        1*4
        [h,p,ci,stats] = ttest(FvFm_uas,FvFm_tas)
```

```
        STE_su = nanstd(FvFm_uao)./sqrt(a); %STE for control score (x axis)
        STE_st = nanstd(FvFm_uas)./sqrt(c); %STE for treatment score (x axis)
        STE_sm = nanstd(FvFm_tao)./sqrt(a); %STE for treatment FvFm (y axis)
        STE_sf = nanstd(FvFm_tas)./sqrt(c);
        x1 = 1;
        x2 = 2;
        x3 = 3;
        x4 = 4;
        y1 = nanmean(FvFm_uao)
        y2 = nanmean(FvFm_uas)
        y3 = nanmean(FvFm_tao)
        y4 = nanmean(FvFm_tas)
        figure
        categories = {'Control','Treatment'};
plot(x1,y1,'c','Marker','diamond','MarkerSize',10,'MarkerFaceColor','c','Mark
erEdgeColor','k')
        hold on
plot(x1,y2,'r','Marker','diamond','MarkerSize',10,'MarkerFaceColor','r','Mark
erEdgeColor','k')
plot(x2,y3,'c','Marker','diamond','MarkerSize',10,'MarkerFaceColor','c','Mark
erEdgeColor','k')
plot(x2,y4,'r','Marker','diamond','MarkerSize',10,'MarkerFaceColor','r','Mark
erEdgeColor','k')%
        hold on
        errorbar(x1, y1, STE_su,'k')
        errorbar(x1, y2, STE_st,'k')
        errorbar(x2, y3, STE_sm,'k')
        errorbar(x2, y4, STE_sf,'k')
        legend('Raw Peaks ','Peaks Smoothed (boxcar)')
        ylabel('Fv/Fm (peak 2)','FontSize',14)
        set(gca,'XLim',[0 3])
        set(gca,'XTick',1:2,'XTickLabel',categories)
        hold off
        if option == 1
            set(gcf, 'PaperPosition',[0.25 2.5 8 6])
            set(gca, 'FontSize',12)
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir 'meta' ext
'_FvFm2.png']);
            set(gcf, 'PaperPosition',[0.25 2.5 4 3])
            set(gca, 'FontSize',8)
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir 'meta' ext
'_FvFm2_th.png']);
        end
        pause
    elseif type ==2
    elseif type == 3
        matname = [name '_concen_ns.mat'];
        Fname = getfname(matname)
        load([FPpamdir Fname])
        ext = '_concen';
        FvFms_o = [metafc(:,15)]; %use WALZ output
        for z = 1:length(FvFms_o)

            if FvFms_o(z,1) < 0
                FvFms_o(z,1) = 0;
            else
                FvFms_o(z,1) = FvFms_o(z,1);
```

```
            end
        end
        FvFm_ao = [FvFms_o(:,1)];
        xaxis = [metafc(:,17)];
        FvFm_uao = (FvFm_ao(find(metafc(:,3) == 1 & metafc(:,16) == 1)))
        x_su = xaxis(find(metafc(:,3) == 1 & metafc(:,16) == 1));
        a = length(FvFm_uao);
        FvFm_uac = (FvFm_ao(find(metafc(:,3) == 1 & metafc(:,16) == 2)))
        x_sc = xaxis(find(metafc(:,3) == 1 & metafc(:,16) == 2));
FvFm_tao = (FvFm_ao(find(metafc(:,3) == 1 & metafc(:,16) == 3)))
        x_st = xaxis(find(metafc(:,3) == 1 & metafc(:,16) == 3));
        b = length(FvFm_tao);
        FvFm_tac = (FvFm_ao(find(metafc(:,3) == 1 & metafc(:,16) == 4)))
        x_stc = xaxis(find(metafc(:,3) == 1 & metafc(:,16) == 4)); %should be
figure
        plot(x_su,FvFm_uao,'.g','MarkerSize',30)
        hold on
        plot(x_sc,FvFm_uac,'.c','MarkerSize',30)
        plot(x_st,FvFm_tao,'.r','MarkerSize',30)
        plot(x_stc,FvFm_tac,'.k','MarkerSize',30)
        xlabel('Concentration Factor')
        ylabel('Fv/Fm (initial) ','FontSize',14)
        legend('Control','Conc. Control. ','Treatment','Conc. Treatment')
        hold off
        if option == 1
            set(gcf, 'PaperPosition',[0.25 2.5 8 6])
            set(gca, 'FontSize',12)
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir 'meta' ext
'_FvFm1.png']);
            set(gcf, 'PaperPosition',[0.25 2.5 4 3])
            set(gca, 'FontSize',8)
            print(['-f' num2str(gcf)],'-dpng',[HTMLpamdir 'meta' ext
'_FvFm1_th.png']);
        end
        FvFms_o = [metafc(:,18)]; %use WALZ output
        xaxis= [metafc(:,19)];
        %uptake and concentrated; treatment and tconcen
        FvFm_uao = (FvFms_o(find(metafc(:,3) == 1 & metafc(:,16) == 1)))
        x_su = xaxis(find(metafc(:,3) == 1 & metafc(:,16) == 1));
        a = length(FvFm_uao);
        FvFm_uac = (FvFms_o(find(metafc(:,3) == 1 & metafc(:,16) == 2)))
        x_sc = xaxis(find(metafc(:,3) == 1 & metafc(:,16) == 2)); %should be
FvFm_tao = (FvFms_o(find(metafc(:,3) == 1 & metafc(:,16) == 3)))
        x_st = xaxis(find(metafc(:,3) == 1 & metafc(:,16) == 3)); %should be
        b = length(FvFm_tao);
        FvFm_tac = (FvFms_o(find(metafc(:,3) == 1 & metafc(:,16) == 4)))
        x_stc = xaxis(find(metafc(:,3) == 1 & metafc(:,16) == 4)); %should be
figure
        plot(x_su,FvFm_uao,'.g','MarkerSize',30)
        hold on
        plot(x_sc,FvFm_uac,'.c','MarkerSize',30)
        plot(x_st,FvFm_tao,'.r','MarkerSize',30)
        plot(x_stc,FvFm_tac,'.k','MarkerSize',30)
        xlabel('Score')
        ylabel('Gain ','FontSize',14)
        set(gca,'XLim',[0 1.4])
        set(gca,'YLim',[0 30])
```